Learning singularity avoidance from constrained motion

Manavalan, Jeevan

Awarding institution:
King's College London

KING'S COLLEGE LONDON

Learning Singularity Avoidance from Constrained Motion

by

Jeevan Manavalan

A thesis submitted in partial fulfillment of the
degree of Doctor of Philosophy

in the

Faculty of Natural & Mathematical Sciences
Department of Engineering

December 2020

Declaration of Authorship

I, Jeevan Manavalan , declare that this thesis titled, 'Learning Singularity Avoidance from Constrained Motion' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Jeevan Manavalan

Date: 06/12/2020

KING'S COLLEGE LONDON

Abstract

Faculty of Natural & Mathematical Sciences

Department of Engineering

Doctor of Philosophy in Robotics

Learning Singularity Avoidance from Constrained Motion

by Jeevan Manavalan

With the continual growth of technology and resulting shift towards the development of versatile and autonomous robots, robotic systems are able to perform increasingly more complex tasks. In turn, expanding their application to other fields. Programming by demonstration is particularly suitable in tasks which require expert knowledge. As novice users can proficiently demonstrate a task in which they are an expert with little or no understanding of the system being taught. The promise of introducing collaborative robots for automation outside traditional manufacturing settings is their usability by novice users, *i.e.,* those potentially with domain-expertise but little knowledge of robotic engineering. It is envisaged that such users would teach task-oriented skills through demonstration, thereby avoiding the need for formal training in the techniques and concepts familiar to roboticists. It has long been established, for example, that *differences in embodiment* of the human musculoskeletal system and most robotic actuation systems mean that direct imitation of human motor behaviour is suboptimal for robots. With this in mind, when designing interfaces and approaches to the programming by demonstration of systems, there is a need to take account of the natural motor behaviour of novices, as well as, manipulability of the taught robot.

This dissertation explores the optimisation of redundancy in robots by learning task-oriented behaviour through programming by demonstration; for the transfer

of demonstrated skills to robotic systems. The research is split over three stages. First, looking at how the constraint learning system copes with demonstration data, namely how it is processed with respect to data dimensionality. A new method is proposed to quickly handle the ever increasing complexity of data provided to systems. It works by reducing the constraint learning system's search space by applying gradient descent. This enables the system's use in learning tasks within shorter periods of time and of greater dimensionality. Now, with the system's propagated applicability to complex problems, the second stage looks at the output of the system. Specifically, how data is prepared for use by the taught robot. Thus, a method is developed to resolve redundancy in learnt task-oriented behaviour. It works by taking the robot's own structure into account subject to a learnt constraint and uses this information to avoid singularities through learnt manipulability maximisation. Finally, the third stage looks at improving how data can be extrapolated when teaching comes from human demonstrators. Thus, a method is proposed which takes stereotypical behaviours in natural human movements into account. Thereby, making use of assumptions on how the demonstrator resolves redundancy, which makes it easier to learn the constraints contained within a task. A pipeline is presented for transferring behaviour optimised for humans, such that these are adapted according to the robotic system's own embodiment. Various experiments are conducted including in the real world which demonstrate the system's applicability to different tasks such as in reaching objectives as well as closing drawers.

Acknowledgements

I would like to thank my supervisor, Dr. Matthew Howard, for his on-going guidance and precious time throughout the PhD, and for offering me the opportunity to delve into my passion of machine learning with humanoids. My particular appreciation is mentioned here for his invaluable feedback during the development of novel algorithms and their stringent evaluations.

Special mention is made of the scholarship awarded by the Department of Engineering at King's College London

I also thank my father, mother and sister for their continued support, and last but not least I would like to mention my special thanks to Élodie Murali for her kind support, especially while I focusing on work and deadlines.

Contents

List of Figures

List of Tables

Abbreviations

CCL	**C**onstraint **C**onsistent **L**earning
DoF	**D**egree of **F**reedom
DPL	**D**irect **P**olicy **L**earning
LUBA	**L**oading on the **U**pper **B**ody **A**ssessment
NIOSH	**N**ational **I**nstitute for **O**ccupational **S**afety and **H**ealth
NMIE	**N**ormalised **M**anipulability **I**ndex **E**rror
NMSE	**N**ormalised **M**ean **S**quared **E**rror
NPPE	**N**ormalised **P**rojected **P**olicy **E**rror
OCRA	**C**oncise **E**xposure
OWAS	**O**vako **W**orking **P**posture **A**nalyisis **S**ystem
PCA	**P**rincipal **C**omponent **A**nalysis
PPE	**P**rojected **P**olicy **E**rror
QEC	**Q**uick **E**xposure **C**heck
REBA	**R**apid **E**ntire **B**ody **A**ssessment
ROM	**R**ange of **M**otion
ROSA	**R**apid **O**ffice **S**train **A**ssessment
RULA	**R**apid **U**pper **L**imb **A**ssessment

Symbols

Below is a list of symbols used throughout this thesis (unless an exception is noted in the text). Note that bold upper-case letters denote matrices, bold lower-case letters denote vectors, and normal letters denote scalars. Notations of the form $\boldsymbol{f}(.)$ denote an argument should be passed to the function $\boldsymbol{f}$.

$\mathbf{x}$	Observed States
$\mathbf{u}$	Observed Actions
$\boldsymbol{\pi}$	Null Space control policy
$\mathbf{b}$	Task Space control policy
$\mathbf{A}^T$	Transpose of matrix $\mathbf{A}$
$\mathbf{A}$	Pfaffian constraint matrix
$\mathbf{A}^\dagger$	Moore-Penrose pseudo-inverse of matrix $\mathbf{A}$
$\mathbf{N}$	Null space projection matrix
$\mathbf{I}$	Identity matrix
$\mathbf{P}$	Projection matrix
$\mathbf{v}$	Task space component
$\mathbf{w}$	Null space component
$\mathcal{P}$	Dimensionality of the state space
Q	Dimensionality of the action space
S	Dimensionality of the task space
t	time
$\mathbf{q}, \dot{\mathbf{q}}$	Angle and velocity of the joint-space

$\mathbf{r}, \dot{\mathbf{r}}$	End-effector position and velocity
$\mathcal{N}$	Number of data points
$\mathbf{U}$	Matrix of Observed Actions $\mathbf{u}_1, \cdots, \mathbf{u}_{\mathcal{N}}$
ϖ	Task space scaling factor
ρ	Null space scaling factor
$E[.]$	Objective or error function. Arguments denote the quantity to be optimised
$\hat{\boldsymbol{\alpha}}$	Unit vector representation of $\boldsymbol{\alpha}$
$\|\|\boldsymbol{\alpha}\|\|$	Magnitude of vector $\boldsymbol{\alpha}$
$\mathcal{L}$	Number of θ
j	number of iterations
k	initial number of θ estimates
$\boldsymbol{L}$	Gain matrix
ω	Step size of θ estimate in gradient descent
μ	Yoshikawa's Manipulability Index
$\boldsymbol{f}(.)^*$	Point attractor target, e.g., ξ^*, ρ^* or $\mathbf{x}^*$
$\tilde{\boldsymbol{f}}(.)$	Estimate of $\boldsymbol{f}(.)$e.g., $\tilde{\boldsymbol{f}}(\mathbf{x})$ is the estimate of $\boldsymbol{f}(\mathbf{x})$ at point $\mathbf{x}$
γ	Normalised task manipulability measure
κ	Fixed time step between observed actions
∇	Gradient
$\boldsymbol{\Lambda}$	Selection matrix
$\boldsymbol{\lambda}$	Elements along the diagonal of $\boldsymbol{\Lambda}$
$\boldsymbol{\Phi}$	Feature matrix, e.g., containing candidate constraints
$\mathbf{J}$	Jacobian matrix
$\boldsymbol{\psi}$	Replaced Null Space control policy
ϵ	Scaling factor for additive white Gaussian noise
$\mathbf{A} \oslash \mathbf{B}$	*Hadamard* (element-wise) *division* of matrix $\mathbf{A}$ by matrix $\mathbf{B}$

Dedicated to my family

For their endless support and encouragement

Chapter 1

Introduction

The promise of introducing collaborative robots for automation outside traditional manufacturing settings is their usability by novice users, *i.e.,* those potentially with domain expertise but little knowledge of robotic engineering. This is being realised with the continual growth of technology and resulting shift towards the development of versatile and autonomous robots. As robotic systems are able to perform increasingly more complex tasks. In turn, expanding their application to other fields. It is envisaged that users of such systems would teach task-oriented skills through demonstration, thereby avoiding the need for formal training in the techniques and concepts familiar to roboticists. Programming by demonstration is particularly suitable in tasks which require expert knowledge. As novice users can proficiently demonstrate a task in which they are an expert with little or no understanding of the system being taught. While there are various ways for systems to learn from demonstrations, the most common approaches include (i) kinaesthetic demonstrations where the system is being manually guided by a person, as well as, (ii) imitation by observing human demonstrations [1]. In the former case, a person can adapt their movements to that of the system which can make it easier for the system to repeat. However, this also makes it more difficult for the demonstrator who may have little to no experience with handling robotic systems (especially if its embodiment is vastly different). In

the latter case, a human can easily demonstrate tasks in a way which they are comfortable with. However, in this case it proves to be more difficult for the learner tasked with making up for the differences in embodiment. Therefore, there is a need to take account of the natural motor behaviour of novices when adapting movements suited for humans. It has long been established, for example, that *differences in embodiment* of the human musculoskeletal system and most robotic actuation systems mean that direct imitation of human motor behaviour is suboptimal for robots [2]. Moreover, in both kinaesthetic guiding and human demonstration observation, when designing interfaces and approaches to the programming by demonstration, the manipulability of the learner needs to be considered. This will help avoid configurations which may lead to undesirable behaviour such as singularities. This suggests the need for *selectivity in imitation*, whereby only task-critical features of behaviour are mimicked, while secondary features–those that are idiosyncratic to the demonstrator–can be replaced. In this case, this is where constraint learning proves particularly useful, as interactions with the environment tend to involve some type of constraint. Therefore redundancy from the demonstrated motions can be separated through a hierarchical task decomposition from a kinematically constrained system, where the movement space is split into a task space and a null space. The task space refers to the degrees of freedom (DoF) required to perform the primary task such as reaching towards a target point and the null space controls a second lower priority objective component.

The focus of this thesis is on learning constraint models from movement data such that generalisation of the task can be achieved across systems of different embodiment, as well as, using the learnt model to allow for redundancy resolution through optimisation of its null space. This optimisation is achieved by replacing null space movement with a control policy which accomplishes a secondary goal such as singularity avoidance without interfering with the task-oriented behaviour. This work is presented over three stages. First, the existing constraint learning algorithm is modified by optimising its search space parameters through the use of gradient descent.

This not only allows for higher DoF systems to be evaluated with lower cost hardware, but also achieves learning in significantly faster times. Second, the so-called *manipulability* analysis, first introduced by Yoshikawa [3], is embedded into the constraint learning approach. This produces a new way to learn a constrained system's manipulability using motion data. The use of this approach is demonstrated by replacing a system's decomposed null space for singularity avoidance through learnt manipulability maximisation. Third, a novel approach is formed where ergonomic priors from human demonstrations are considered to help learn constraints contained within tasks. This approach is shown to be used for the transfer of generalised task-oriented behaviour to systems of a different embodiment. This transfer is done in a way such that the learnt behaviours can be adapted to suit the provided system without explicit knowledge of the the constraints. All of these approaches are validated through extensive simulated and real world experiments. The real world experiments demonstrate the system's applicability to various tasks such as in reaching objectives as well as closing drawers.

1.1 Thesis Outline

In this section, a brief outline of the thesis is given. The chapters are described by their key contents as well as their relating publications (see Section 1.2 for a list of publications).

Chapter 2 reviews the state-of-the-art in learning from demonstration data and modelling thereof. Ergonomics also plays a vital role as programming by demonstration is not only restricted to kinaesthetic demonstrations using robotic systems. Instead, it can also be performed by observing human motions, which is one of the key procedures to a novel method proposed later on. Finally, looking at optimisation of systems using learnt behaviours. Although it is possible to consider many different types of optimisation, this research primarily focuses on manipulability. As its maximisation can be used to avoid singularities, which could otherwise have a catastrophic effect if learnt behaviours are not properly adapted to the embodiment of the learner. A guide on constraint learning (presented in the literature review and throughout the thesis) which shows how to make use of the Constraint Consistent Learning (CCL) open source software library is **published in [III]**

Chapter 3 proposes an approach to mitigate issues surrounding learning in increasingly complex tasks, more specifically data dimensionality. Gradient descent is applied to the, at the time, state-of-the-art constraint learning approach in order to reduce the search space when learning constraints. The approach can be applied to systems subject to unknown constraints and works without any prior information regarding the dimensionality of the constraints. The work is followed by an evaluation of the extent to which gradient descent optimises learning performance. The evaluations show how the prior state-of-the-art is limited to 4D data by the hardware used in the experiments. However, using gradient descent allows for learning on 9D data under the same hardware limitations. This proposed approach makes constraint learning applicable to modern robotic systems with greater degrees of

freedom (DoF) and significantly reduces the time taken to learn a constraint model. As it also has a lower demand on computation costs, this in turn expands its applicability to lower cost computers. **Published in [I]**.

Chapter 4 shows a new approach to learning the manipulability of constrained systems from demonstration data. It works without explicit knowledge of the dimensionality of constraints nor underlying control policies. This work considers the control of systems subject to uncertain constraints from a set of candidate constraints, due to the complexity and/or naivety of non-expert users. Using this approach, a system can perform task-oriented behaviour with a replaced and optimised null space control policy. The replaced policy can use the learnt constrained manipulability for manipulability maximisation. Through this approach, redundancy in the system is resolved through singularity avoidance which is learnt from the demonstrations. Extensive experiments are conducted showing its suitability to systems in both the joint space and end-effector space, as well as, performance comparisons to different null space policies. Moreover, tests are performed to learn linear and non-linear constraints. Real world experiments are also presented using a 7DoF robot in a reaching task as well as a drawer closing task. **Published in [II]**.

Chapter 5 discusses a new method for learning task constraints directly from human demonstrations based on stereotypical features of demonstrators' posture control. It learns the task and null spaces involved in the behaviour and their underlying constraints. Thereby, enabling the decomposition of task-oriented motions from demonstrations, such that these can be used by a system of a different embodiment. The approach plays a significant role for use by naive users as demonstrations are performed by the human in a natural manner without any interaction with the robot. The robot is then able to repeat the task whilst adapting its redundancy resolution in a way suitable to its own structure, rather than that of the demonstrator. The approach accounts for differences in human ergonomics and robot manipulability where both the demonstrator and learner may have different optimal poses,

i.e., what is ergonomic for a human may lead to a singular posture in a robot. This approach uses knowledge from ergonomic literature to form an estimate of how humans resolve redundancy which can be used to decompose task-oriented motions for its transfer to robotic systems. The focus of this work lies in making it as natural as possible for a human to demonstrate a task, as it cannot be expected for naive demonstrators to have any understanding of handling robots. Following this, a pipeline is presented which takes natural demonstrations from a human and generalises the task-oriented behaviour for its execution on a robotic system of a different embodiment. Experiments are presented which include learning constraints in absence of prior knowledge, as well as, learning with candidate rows which allows for greater complexity in tasks using significantly lesser data. Finally, it is tested in the real world in a drawer opening/closing task with data recorded from human demonstrations. The constraint is learnt and the decomposed task-oriented trajectory is reproduced on a 7DoF robot with a different null space control policy subject to the same constraints.

Chapter 6 provides a conclusion to the research performed throughout this thesis and proposes directions for future work.

1.2 Publication Summary

Publications

[I] Jeevan Manavalan and Matthew Howard. Learning null space projections fast. In *European Symposium on Artificial Neural Networks (ESANN)*, 2017.

[II] Jeevan Manavalan and Matthew Howard. Learning singularity avoidance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[III] Jeevan Manavalan, Yuchen Zhao, Prabhakar Ray, Hsiu-Chin Lin, and Matthew Howard. A library for constraint consistent learning. *Advanced Robotics*, 34:845–857, 2020.

Chapter 2

Literature Review

2.1 Introduction

This chapter discusses relevant background to the research presented throughout this dissertation. Section 2.2 provides a general overview of imitation learning and looks at different types of ways to approach motion modelling. As with many other topics, policy derivation to create models is very vast with an abundance of surveys and well founded literature [1, 4–6]. Although a variety of approaches are explained in this section, an in-depth survey detailing a wider range of approaches is available in [6] and [4]. Section 2.3 discusses human ergonomics which plays a vital role in Chapter 5 when optimising learning to work more robustly with human demonstrations. Popular risk assessments to evaluate a teachers posture during demonstrations are analysed. Finally, Section 2.4 looks at kinematic singularities and ways to avoid these which relates closely to research conducted in Chapter 4.

2.2 Statistical Motion Modelling and Imitation

Statistical motion modelling and imitation are a core concept to handling control problems in robotics [1, 4, 5]. Its popularity comes from it being one of the more natural and intuitive ways to teach a system by providing demonstrations which exhibit desired behaviour for the learner to emulate [4]. Imitation learning is closely related to reinforcement learning, *i.e., learning a policy to solve a problem through trial and error by maximising the expected reward at each time step using a reward function which is discounted over time by some factor.* However, one of the main advantages of imitation learning is how it handles increasingly complex environments such as with assistive robots, self-driving vehicles and human computer interaction. Typically in such cases, there are far too many possible scenarios to take account of when optimising behaviours through trial and error which makes reward functions very difficult to define. This difficulty in finding an appropriate reward function correlates to the exponentially growing complexity of a system's interaction with its environment, therefore it is generally the consensus that having prior knowledge of a task imparted by an expert is a far more efficient way to have a system learn [6–9]. Further to this, it is understandably a lot easier for novice users, *i.e.,* those potentially with domain expertise but little knowledge of robotic engineering, to simply demonstrate a task rather than to express it algorithmically for the required knowledge to be transferred onto a system [10].

By making use of demonstration data such as motion data captured from kinaesthetic demonstrations, features contained within this data can be extracted. This in turn can be applied to new situations or systems to exhibit certain behaviours. While there are various ways to collect such data, the most common approaches include (i) directly from human demonstrations or (ii) a robot that is either moving by itself or manually being guided by a person [1].

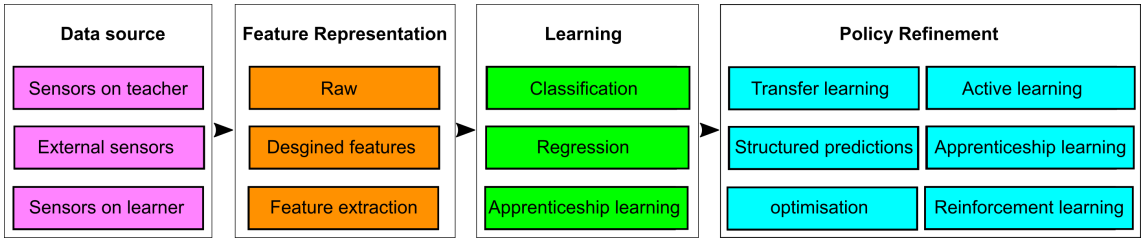


FIGURE 2.1: Imitation learning flowchart. Adapted from [4].

Figure 2.1 shows a generalised workflow of the main stages in imitation learning. It all starts with collecting data using sensors which can be on the teacher such as during human demonstrations. This approach is intuitive for the teacher depending on the type of sensors used as not to restrict the movements in any way. However, it brings a necessity to adapt the obtained data afterwards to suit the learner. Sensors can also be placed on the learner, for example when a robot is being manually guided. This can be less intuitive for the teacher, who may have little to no experience in handing that robot. However, once the teacher successfully guides it to collect enough data, this is usually already adapted to the embodiment of the learner. Finally, sensors can be located externally, such as by using a camera or some other peripheral for recording, making it easy for the demonstrator to perform comfortably. However, this requires a greater emphasis on processing the data to extract relevant information from within the data set. Moreover, the data set is typically of a higher dimensionality than what is required for the task. On top of this, the data in this case still needs to be modified to fit the learner. Once data is collected, in some cases the raw data is directly provided for training a model. At other times, functions can be developed which make use of expert knowledge to identify what is relevant to the task within the given data set. Data can also be processed by the extraction of features, such as for mapping onto a lower dimensional state space using common approaches like principal component analyisis (PCA). Once the data is in a suitable form to perform learning, a popular way to do so is by using a classifier where observations are automatically placed into a finite set of groups. Alternatively, regression can be used if the observed actions are in a continuous space. In this case,

the main difference being that the mapping of input states are to a numeric output based on the action, which makes regression very suitable to low level motor actions. Another way to learn as well as to refine a model is through apprenticeship learning (also referred to as inverse reinforcement learning). Using this approach, samples of demonstrations which produce the desired actions are used as a template to learn the reward functions, which in turn are optimised by an expert to improve the model. This approach is particularly useful when no clear reward function can be defined. Refining the learnt model is a step which is not necessarily limited to post-learning (as shown with apprentice learning). In fact, refining a policy through a feedback loop during the learning process is a very common approach. This refining process can be achieved in various ways such as through reinforcement learning (defined earlier in this section) or using active learning. Active learning is where uncertainty for the mapping of some states to a model can be resolved by querying an expert for the best solution. It plays an important role to adapting a model to situations which are not provided in training data. Alternatively, transfer learning can be used where pre-existing knowledge can be converted from another task or other agents to aid in refining the model. This is particularly beneficial when it comes to avoiding the need to obtain new samples, when instead pre-existing information can be adapted saving time and/or cost. Structured predictions is another way to handle policy improvement based on relating actions to previous states. While this approach can be used to iteratively optimise a model at each step of an observed action, errors leading to an unseen state can have a compounding deteriorative result. Finally, refining can be performed through optimisation techniques, by typically generating a random model which is iteratively improved according to some fitness function, with the goal of finding input parameters which minimise said function [4].

2.2.1 Direct Policy Learning

One of the most fundamental approaches to modelling movements is through Direct Policy Learning (DPL) [1, 5, 11]. Given data of observed movements as states $\mathbf{x}$ and actions $\mathbf{u}$, this approach, usually formulated as a supervised learning problem, estimates a function of the policy $\boldsymbol{\pi}$ through direct regression, which is a mapping between states and actions

$$\mathbf{u} = \boldsymbol{\pi}(\mathbf{x}), \boldsymbol{\pi} : \mathbb{R}^{\mathcal{P}} \rightarrow \mathbb{R}^{\mathcal{Q}}$$

where $\mathbf{x} \in \mathbb{R}^{\mathcal{P}}$ and $\mathbf{u} \in \mathbb{R}^{\mathcal{Q}}$ are the state and action spaces, respectively.

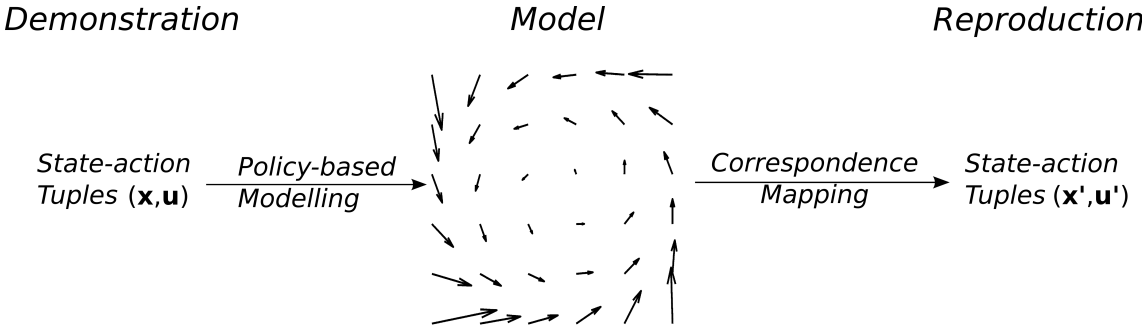


FIGURE 2.2: Policy based modelling. Demonstration data is given as pairs of states and actions which are used to form a model of the policy represented here as a vector field. To then reproduce the movement from the resulting model, a corresponding mapping is defined which maps the demonstrators states $\mathbf{x}$ and actions $\mathbf{u}$ to the imitators $\mathbf{x}'$ and $\mathbf{u}'$ [12].

Once demonstration data is used in learning, this results in a reproduced movement by an imitator as illustrated in Figure 2.2. As shown, data is given as pairs of states and actions. A model of the policy is then produced which results in corresponding pairs of state-dependent actions, with the goal being to approximate the policy as close as possible. One way to obtain a mapping between the states and actions is

by minimising the error of

$$E_{DPL} = \sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\boldsymbol{\pi}}_n||^2$$

where $\tilde{\boldsymbol{\pi}}$ is an estimated function of the policy $\boldsymbol{\pi}$.[1]

As pointed out in [11], when learning policies through DPL, demonstrations are either performed in free space such as doing sign language or under consistent constraints such as interacting in an unchanging environment with obstacles. These two categories that fall under demonstrations that follow DPL are referred to as *unconstrained* or *consistently constrained*, respectively. Moreover, DPL being more of a traditional and older planning algorithm, is deterministic in nature and suffers from poor generalisation when dealing with undemonstrated states. It simply assumes that each action leads to a particular state, however this ideology does not always apply in the real world due to the ever changing environment [1]. Nonetheless, there are extensions to this approach to mitigate its lack of robustness to generalisation. One approach is through exploration policies which typically implements a reward function, yet this too brings up another issue involving finding a balance between *Exploration vs. Exploitation*, *i.e.,* deciding how much more varying data is required compared to being satisfied with the policy modelled up to a certain point in time. The risk lies in excess time spent at improving a policy not justified by the rate of improvement of said policy. Ultimately, other approaches exist which focus on generalisation without the same shortcomings.

[1]For brevity, here, and throughout the thesis, the notation $\mathbf{a}_n$ may be used to denote the quantity $\mathbf{a}$ evaluated on the nth sample. For example, if $\mathbf{a}$ is a vector quantity computed from the state $\mathbf{x}$, then $\mathbf{a}_n = \mathbf{a}(\mathbf{x}_n)$.

2.2.2 Constraints & Constraint Policy

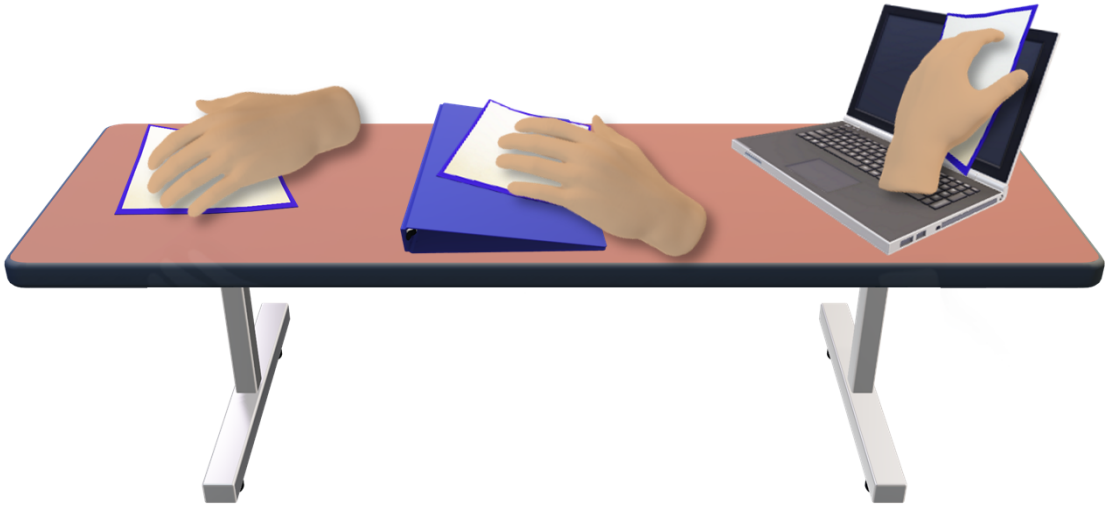


FIGURE 2.3: System restrictions on different environmental constraints.

As discussed in Section 2.2.1, there are implicit assumptions set by DPL regarding demonstrations such that these are expected to be either *unconstrained* or *consistently constrained* [11]. However, this makes it infeasible when it comes to modelling many everyday tasks due to variability in the real world. Interactions with the ever changing environment tend to involve some type of constraint subject to variability. This may have an impeding factor to the success of a task, such as when walking on uneven terrain or how a surface of a table limits movement of a hand at various angles during a wiping task (Figure 2.3) [13]. Not being able to take such variability into account which should be expected in a real world environment shows a limitation of DPL [11].

While constraints are generally viewed as an impeding factor to a task, they can also be used to aid systems in interactions such as by exploiting environmental constraints in order to help facilitate grasping [14–17]. For example, supporting surfaces such as floors and tables, can be used to assist in grasping as demonstrated in a slide-to-edge grasping strategy (Figure 2.4-A below) and a force-compliant grasping method (Figure 2.4-B below).

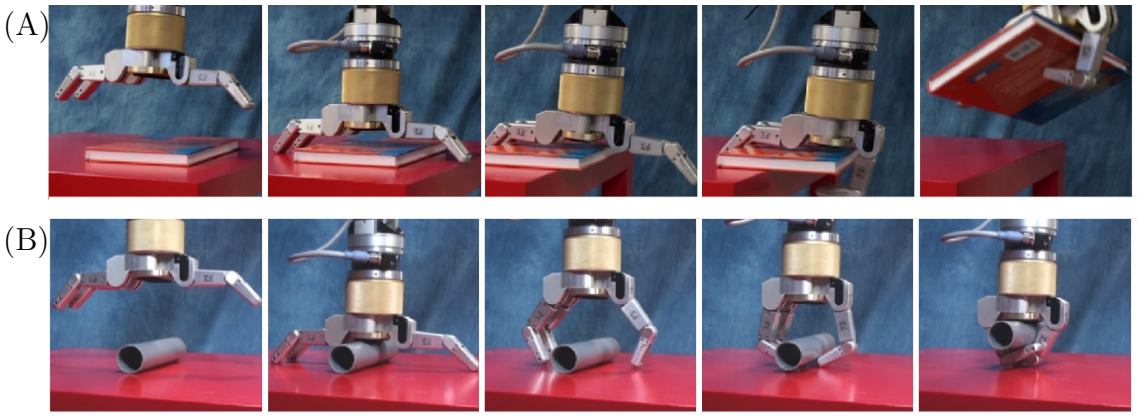


FIGURE 2.4: Exploiting environmental constraints to aid in grasping. (A) shows the Slide-to-edge grasp strategy where an object is moved along a surface until it is partially over an edge so that the system can grasp it from underneath. (B) shows a Force-compliant closing strategy where opposing fingers move towards each other along a surface so that they can slide under a target object which lies in centre of the grasp. Extracted from [17].

Complex interactions can also be derived from self-imposed constraints, such as pouring water from a bottle, as the hand's orientation is restricted such that the water is poured into the glass. Essentially, robotic manipulators are expected to produce a set of joint-space movements which comply with the constraint [11, 18]. Based on the principles of analytical dynamics, skills can be decomposed into an unconstrained policy and the constraints [19]. Starting with the DLP based approach (Section 2.2.1) which describes autonomous systems of the form

$$\mathbf{u}(t) = \boldsymbol{\pi}(\mathbf{x}(t)), \qquad \boldsymbol{\pi} : \mathbb{R}^{\mathcal{P}} \in \mathbb{R}^{\mathcal{Q}} \tag{2.1}$$

where demonstration data is given as $\mathcal{N}$ pairs of observed states $\mathbf{x} \in \mathbb{R}^{\mathcal{P}}$ (usually represented either in end-effector or joint space) and actions $\mathbf{u} \in \mathbb{R}^{\mathcal{Q}}$, and $\boldsymbol{\pi}$ is a direct mapping between the two. When including a constraint, the model is subject to a set of $\mathcal{S}$-dimensional constraints where $\mathcal{S} < \mathcal{Q}$

$$\mathbf{A}(\mathbf{x})\mathbf{u}(\mathbf{x}) = \mathbf{b}(\mathbf{x}) \tag{2.2}$$

$\mathbf{A}(\mathbf{x}) \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ is a matrix describing the constraints and $\mathbf{b}(\mathbf{x}) \in \mathbb{R}^{\mathcal{S}}$ is some vector of the *task space policy* describing the primary task to be accomplished. The constraints project the actions of the policies onto the null space of the constraints. By inverting Equation 2.2, observed actions can be described as

$$\mathbf{u}(\mathbf{x}) = \underbrace{\mathbf{A}^{\dagger}(\mathbf{x})\mathbf{b}(\mathbf{x})}_{\mathbf{v}} + \underbrace{\mathbf{N}(\mathbf{x})\boldsymbol{\pi}(\mathbf{x})}_{\mathbf{w}} \tag{2.3}$$

$\mathbf{A}^{\dagger} = \mathbf{A}^{\top}(\mathbf{A}\mathbf{A}^{\top})^{-1}$ is the Moore-Penrose pseudo-inverse of $\mathbf{A}$. As previously mentioned, the Moore-Penrose pseudo-inverse is applicable in this case as $\mathbf{A}(\mathbf{x}) \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ is subject to a set of $\mathcal{S}$-dimensional constraints where $\mathcal{S} < \mathcal{Q}$. If $\mathbf{A}$ has full rank, which allows for $\mathbf{A}^{\dagger} = \mathbf{A}^{-1}$, this indicates a fully unconstrained system which does not satisfy $\mathcal{S} < \mathcal{Q}$ and is therefore not considered.

$$\mathbf{N}(\mathbf{x}) := \mathbf{I} - \mathbf{A}(\mathbf{x})^{\dagger}\mathbf{A}(\mathbf{x}) \in \mathbb{R}^{\mathcal{Q}\times\mathcal{Q}} \tag{2.4}$$

is the null space projection matrix that projects the null space policy $\boldsymbol{\pi}(\mathbf{x})$ onto the null space of $\mathbf{A}$. Here, $\mathbf{I} \in \mathbb{R}^{\mathcal{Q}\times\mathcal{Q}}$ denotes the identity matrix. $\mathbf{v}$ is the task space component that implements the task space policy and $\mathbf{w}$ is the null space component. The task space can refer to the degrees of freedom (DoF) required to perform the primary task such as reaching towards a target point, and the null space controls a second lower priority objective in a way where it doesn't interfere with the main task such as avoiding joint-limits, self-collision or kinematic singularities [20, 21]. This formalism is generic and works for a wide variety of systems, it not only applies to kinematics, but also to redundant actuation [22], and redundancy in dynamics [19]. Constrained systems described by this form (Equation 2.3) can appear in various scenarios such as when interacting with physical objects e.g. wiping a table where the surface of the table acts as a constraint on the system [23], which can be referred to as contact constraints [24]. Moreover, these constraints can also be non-linear such as curved surfaces discussed in Section 2.2.3. Section 2.2.3 identifies

the concepts of state-independent and state-dependent constraints, which were first introduced in [13] and is shown in Figure 2.7 (discussed later). It is often the case that manipulators contains a high level of redundancy with the DoF available to execute a task usually being higher than what is actually required [25]. For example, in a task demonstrated by a human where the target objective is to keep a finger on a particular point, there is redundancy in the position of the elbow allowing it to be flared out at various degrees while still satisfying the main goal. To put this in context to the aforementioned formalism, setting $\mathbf{A}$ such that it maps the Jacobian from the joint-space to end-effector position, and setting $\mathbf{b} = 0$, would result in the end-effector, in this case being the fingertip, to remain still while allowing flaring of the elbow which doesn't interfere with the stationary fingertip.

2.2.2.1 Constraint in the Joint-Space

Following on from the constraint learning policy, its representation of the formalism in a kinematic setting can be as follows, where we consider the state and actions as joint-angles and joint-velocities, $\mathbf{x} \equiv \mathbf{q}$ and $\mathbf{u} \equiv \dot{\mathbf{q}}$, respectively. In this case, the constraint model will be defined as $\mathbf{A}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{b}(\mathbf{q})$ and accordingly the observed joint-velocities subject to the constraints are

$$\dot{\mathbf{q}} = \mathbf{A}^{\dagger}(\mathbf{q})\mathbf{b}(\mathbf{q}) + \mathbf{N}(\mathbf{q})\boldsymbol{\pi}(\mathbf{q}) \tag{2.5}$$

This control scheme allows for posture control of the system by applying the constraint to its joint-space.

2.2.2.2 Constraint in the End-effector Space

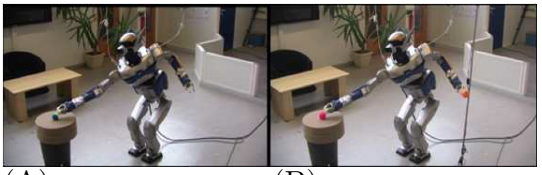
Alternatively to posture control (see Section 2.2.2.1), another way to manipulate a system is through its end-effector space $\mathbf{r}$. Given the states $\mathbf{x} \equiv \mathbf{q}$ and $\mathbf{u} \equiv \dot{\mathbf{q}}$, the task space target $\mathbf{b}(\mathbf{x}) \equiv \dot{\mathbf{r}}(\mathbf{x})$ describes the velocity of the system's end-effector

when moving in some direction subject to the task constraints, for example if the function to obtain $\mathbf{b}(\mathbf{x})$ is set up as a point-attractor, *i.e.,* $\mathbf{b}(\mathbf{x}) = \rho(\mathbf{r}^* - \mathbf{r}(\mathbf{x}))$ where ρ controls the speed of reaching the target point $\mathbf{r}^*$, as it approaches said target the system's end-effector velocity will slow down relative to how far away it is.

2.2.3 Constraint Learning Approaches

As mentioned in Section 2.2.2, the application of learning approaches [26] that do not consider the composition of data in terms of the constraints are prone to poor performance and modelling errors. For example, applying direct regression to learn policies where there are variations in the constraints can result in model averaging effects that risk unstable behaviour [11]. This is due to factors such as (i) the *non-convexity* of observations under different constraints, and (ii) *degeneracy* in the set of possible policies that could have produced the movement under the constraint. However, increasingly more methods are being developed which overcomes the aforementioned issues. These include methods which learn what movements can be performed in the task space [27, 28], as well as, others which separate the task and null space component [29] to then learn the constraint from either the task space [30] or null space [13, 23, 31].

In [27], a method is introduced to learn task prioritised behaviour from a predefined set of possible tasks which can be executed in parallel. The main drive behind this work is to be able to tell apart motions which visually produce similar postures but are performed to achieve a different purpose, for example in Figure 2.5-A the robot is grasping an object with the right arm while maintaining balance using the left one, whereas in Figure 2.5-B the robot is seen in a similar pose but in this case it is using the left arm to hold onto a second ball. It is not



(A) (B)

FIGURE 2.5: Similar postures in different tasks. Extracted from [27].

possible to tell these actions apart by simply looking at the postures. Thus, the motions leading up to this pose needs to be taken into account so that it becomes possible to identify that the robot is doing something different despite its similar pose. It works given a set of pre-defined candidate tasks referred to as *task pool* and their associated controllers. The approach learns tasks by projecting timewise split observed motions onto iteratively selected tasks of the task pool. A cost function assesses the fit of the motion to the selected task. A perfect fit would result in no movement due to cancellation of the motion from the task space and its identical projection onto the null space. This work presents a very distinct approach to learning constraints and is primarily applied on the differentiation of parallel tasks which contain similar postures such as bimanual operations [32].

Handling task prioritisation is also looked at in [33], where different tasks are learnt by evaluating the variability of demonstrations. It works by extracting consistent task space features, so that learnt behaviours can be generalised to new scenarios. In this approach, motion is represented as a spring-damper system which uses gaussian mixture regression for learning. The overall approach can be decomposed into several steps which go over statistical modelling, dynamical systems and optimal control. One of the main limitations pointed out by the authors is that it initially requires a set of candidate frames (*i.e.,* list of coordinate systems) selected by the experimenter to help identify what is relevant to the task. These candidate frames can be subject to overspecification when being selected, resulting in requiring a greater amount of demonstrations to be able to phase out irrelevant frames. The main area to improve here lies in detecting redundancy in frames as well as their correlation to each other. These factors become increasingly important when wanting to determine the extent to which particular frames may contribute to achieving a task. Moreover, the amount of variations required from the demonstrator correlates to the number of candidate frames. This implies that the experimenter needs a good understanding of what is relevant to learning. However, frames can also consist of straightforward candidates such as a list of objects or landmarks. This can to some extent mitigate what would

otherwise be a steep learning curve for novice users. As pointed out in [33] and [34], this approach is widely applicable to various problems as it can deal with task space motions, constraints and priority constraints such as in bimanual tasks in the joint space.

In [23], a method which learns the kinematic constraints by observing movement is presented. Being able to learn constraints purely through movement data overcomes the need for force sensor measurements. Moreover, the approach does not require any information regarding geometry, constraint dimensionality or any information regarding the control policy of the movement. It learns the constraints as a null space projection matrix corresponding to the system's constraints. This approach initially devised from [11] has led to the development of other constraint learning methods which work on similar principles including [13] and [30]. All of these can directly be applied to policies of the form in Equation 2.2 where $\mathbf{b} = 0$. However, if the observed actions contain movement in both the task and null space, it needs to first be separated using the method defined in [29]. This is done by seeking an estimate $\tilde{\mathbf{w}}$ that minimises

$$E[\tilde{\mathbf{w}}] = ||\tilde{\mathbf{P}}\mathbf{u} - \tilde{\mathbf{w}}||^2 \tag{2.6}$$

where $\tilde{\mathbf{P}} := \tilde{\mathbf{w}}\tilde{\mathbf{w}}^\top / ||\tilde{\mathbf{w}}||^2$. This works on the assumption that, there exists a projection $\mathbf{P}$ for which $\mathbf{P}\mathbf{u} = \mathbf{P}(\mathbf{v}+\mathbf{w}) = \mathbf{w}$. This can be used to estimate $\mathbf{v}$ *i.e.,* $\tilde{\mathbf{v}} = \mathbf{u}-\tilde{\mathbf{w}}$. This approach decomposes the movements into orthogonal components $\mathbf{u} \equiv \mathbf{v} + \mathbf{w}$ where variations in the task space policy are phased out against a consistent null space policy.

In [23], the $\mathbf{N}$ projection matrix is learnt given $\mathcal{N}$ pairs of observed states $\mathbf{x} \in \mathbb{R}^{\mathcal{P}}$ and actions $\mathbf{u} \in \mathbb{R}^{\mathcal{Q}}$ subject to various conditions. Moreover, $\mathbf{u} = \mathbf{N}\boldsymbol{\pi}$, which means that $\mathbf{u}$ is obtained by projecting the vector $\boldsymbol{\pi}$ onto the image space of $\mathbf{N}$, but the projection of $\mathbf{u}$ also lies in the image space of $\mathbf{N}$, thus

$$\mathbf{N}\mathbf{u} = \mathbf{u} \tag{2.7}$$

$\mathbf{N}$ can be obtained by finding an estimate which satisfies equation 2.7. Therefore, an estimate of $\tilde{\mathbf{N}}$ is suggested which minimises the difference between $\tilde{\mathbf{N}}\mathbf{u}$ and the raw observations of $\mathbf{u}$:

$$E[\tilde{\mathbf{N}}] = \sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\mathbf{N}}\mathbf{u}_n||^2 \tag{2.8}$$

Expressing this through $\tilde{\mathbf{A}}$ gives:

$$E[\tilde{\mathbf{N}}] = \sum_{n=1}^{\mathcal{N}} \mathbf{u}_n^T \tilde{\mathbf{A}}^\dagger \tilde{\mathbf{A}} \mathbf{u}_n \tag{2.9}$$

To put this in context of a very simple demonstration, consider a system that has a one-dimensional constraint and two DoF, i.e. $\mathbf{A} \in \mathbb{R}^{1x2}$ where $\mathbf{A}$ is a set of $\mathcal{S}$ vectors

$$\tilde{\mathbf{A}} = (\boldsymbol{\alpha}_1^T \boldsymbol{\alpha}_2^T ... \boldsymbol{\alpha}_{\mathcal{S}}^T)^T \tag{2.10}$$

In this case, $\boldsymbol{\alpha}$ refers to the unit vector which can be seen below in a polar coordinate system:

$$\tilde{\boldsymbol{\alpha}} = (cos\theta, sin\theta) \tag{2.11}$$

To get an estimate of $\tilde{\mathbf{A}}$, an optimisation technique called line search is used where a set of $\mathcal{L}$ sample θ are generated and tested for the minimum outcome when applied to equation 2.9. This search is limited to a range between $0 \leq \theta \leq \pi$ as all cases

between π and 2π are also covered in the aforementioned range. Using this brute-force line search on an $\mathcal{L}$-sided grid with a $(\mathcal{Q}-1)$ dimensional space of parameters for selecting $\boldsymbol{\theta}$ results in a sequential running time of $O(\mathcal{L}^{\mathcal{Q}-1})$.

To adapt the problem to handle more DoF, i.e. $A \in \mathbb{R}^{1\times\mathcal{Q}}$ where $\mathcal{Q} > 2$, vectors in $\tilde{\boldsymbol{\alpha}}$ needs to be extended such that it can be represented by $\mathcal{Q}-1$ parameters. To handle a higher dimensional problem, such as $\mathcal{S} = 4$, the vector $\tilde{\boldsymbol{\alpha}}$ needs to be represented by three angles $(\theta_1, \theta_2, \theta_3)$. Figure 2.6 shows a table with representations of α under different dimensions and with different DoF.

	$\mathbb{R}^2$	$\mathbb{R}^3$	$\mathbb{R}^4$
$\hat{\alpha}_1$	$\cos\theta_1$	$\cos\theta_1$	$\cos\theta_1$
$\hat{\alpha}_2$	$\sin\theta_1$	$\sin\theta_1\cos\theta_2$	$\sin\theta_1\cos\theta_2$
$\hat{\alpha}_3$	-	$\sin\theta_1\sin\theta_2$	$\sin\theta_1\sin\theta_2\cos\theta_3$
$\hat{\alpha}_4$	-	-	$\sin\theta_1\sin\theta_2\sin\theta_3$

FIGURE 2.6: Example representation of unit vectors with different DoF against different dimensions [23].

Estimating $\tilde{\mathbf{N}}$ follows the same aforementioned approach which obtains the unit vector $\tilde{\boldsymbol{\alpha}}_s$ for every $\mathcal{S}$ constraint, and seeking the ideal parameters for $\boldsymbol{\theta}_s = (\theta_{s,1} \quad \theta_{s,2} \quad \theta_{s,3}...\theta_{s,\mathcal{Q}-1})^T$ which gives the smallest value with equation 2.9 [23]

When dealing with multidimensional constraints, learning can be performed iteratively where you have several constraint vectors $\tilde{\boldsymbol{\alpha}}_s$, where the $(s+1)^{th}$ vector is added as long as it does not reduce the fit under equation 2.9. One of the benefits to using this approach to learn the projection matrix $\mathbf{N}$ for multidimensional constraints is that the constraints can be decomposed into a set of unidimensional projections:

$$\mathbf{N} = \mathbf{N}_1\mathbf{N}_2...\mathbf{N}_{\mathcal{S}} \tag{2.12}$$

or with the $\boldsymbol{\pi}$ component it equally is

$$\mathbf{u} = \mathbf{N}\boldsymbol{\pi} = \mathbf{N}_1(\mathbf{N}_2...(...\mathbf{N}_{\mathcal{S}}\boldsymbol{\pi})...)) \tag{2.13}$$

This shows that a good approximation of $\tilde{\mathbf{N}}$ can be made by

1. Finding the best $\tilde{\boldsymbol{\alpha}}_1$(*i.e.,* $\boldsymbol{\theta}_1^*$) for equation 2.9 based on the fitting procedure explained earlier.
2. Finding the best $\tilde{\boldsymbol{\alpha}}_2$(*i.e.,* $\boldsymbol{\theta}_2^*$) under the condition that $\tilde{\boldsymbol{\alpha}}_1$ is perpendicular to $\tilde{\boldsymbol{\alpha}}_2$
3. Repeating this process with each remaining constraint $\tilde{\boldsymbol{\alpha}}_{s+1}^*$ until equation 2.9 fails to reduce the error.

A condensed set of steps is presented in algorithm 1.

Algorithm 1: Projection Matrix Learning

Input: State, action samples$\{\mathbf{u}_n\}_{n=1}^{\mathcal{N}}$
Output: $\tilde{\boldsymbol{\alpha}}_i$: the set of constraint vectors

1: Estimate $\tilde{\boldsymbol{\alpha}}_1$ by minimising equation 2.9. Set $s \leftarrow 1$.
2: **while** $E[\tilde{\mathbf{N}}]$ in equation 2.9 is not increasing **do**
3: $s \leftarrow s + 1$
4: Learn $\tilde{\alpha_s}^*$ minimising equation 2.9 such that $\tilde{\alpha_s}^* \perp \tilde{\alpha_i} \forall i < s$
5: Set $\tilde{\mathbf{A}} \leftarrow [\tilde{\boldsymbol{\alpha}}_1{}^T, ..., \tilde{\boldsymbol{\alpha}}_s{}^T]^T$
6: **end while**
7: Return $\tilde{\boldsymbol{\alpha}}_i$

Over the last several years, this approach has been further developed to handle a wider variety of constraints categorised as *state independent or state dependent* [13]. For example, considering a case where the state-space is represented as end-effector coordinates such as when wiping a table (see Figure 2.7-A below), the flat surface acts as a hard restriction on the actions available (motions perpendicular to the surface will be eliminated by the constraint). This restriction applies regardless of where the end-effector is located on the surface, thus it represents a state independent

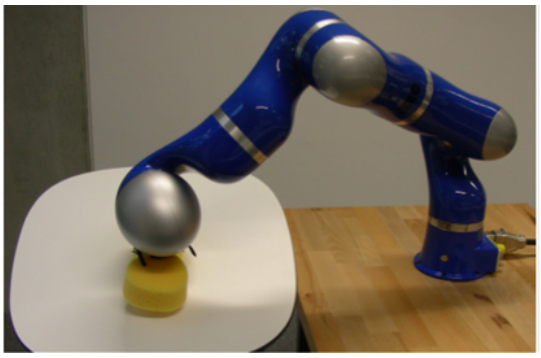
(A)  (B) 

FIGURE 2.7: Comparing state-independent and state-dependent constraints. (A) Table wiping task subject to a state-independent flat environment constraint using a robot arm. (B) A bowl wiping task subject to a state-dependent curvature constraint.

constraint. On the other hand, when wiping a bowl or stirring soup in it (see Figure 2.7-B), the curved surface introduces a state dependency in the constraint. This is because the restriction of motion (in this case being the angle of the bowl's curvature) is dependent on the location of the end-effector. Finally, in [30], a similar approach shows how the constraint matrix $\mathbf{A}$ can be learnt directly from the null space component $\mathbf{v}$ without having to first obtain its null space projection $\mathbf{N}$.

2.3 Ergonomics

Ergonomics is the study of interactions between people and systems[2] [35, 36], with a view to improving and optimising these interactions. These studies provide direction as to what can be added to interactions that contribute towards its optimisation, as well as, what factors degrade it and should be removed. An emphasis towards improving ergonomics can lead to several benefits, for example in the work environment, tasks can be made easier for humans to perform reducing the probability of them making errors during its execution, which in turn can also make it safer to perform. Moreover, the field of ergonomics also provides guidance on customising tasks which consider characteristics or different needs to accommodate variability in humans. As an example, chairs with adjustable heights account for differences in peoples bodies (Figure 2.8) or providing modified tools for use by left-handed people. The effects of optimisation through ergonomics can generally be categorised into either *information-processing* or *manual handling* [35]. In the former, this can be redesigning an interface such that less thought is required by a user, for example when developing new systems while ensuring that they resemble tasks that people can associate with or are already familiar with. In the latter, this focuses on reducing strain on the musculoskeletal system and can be

FIGURE 2.8: Example of ergonomics in chair design. Left is a folding chair prioritising space and short periods of use. Right is a chair designed with ergonomics in mind, as seen in the adjustable height, curved back to support the natural 'S' shape of the spine, swivel bottom for increased manoeuvrability and arm rests which allow the shoulders to relax.

[2]Systems by this definition and in the context of ergonomics is used more broadly to not only refer to robotic systems, hardware and software, but also includes tasks, jobs, products and environments.

as simple as adding handles to a drawer to make it easier to use. A large focus of ergonomics falls upon improving performance, reducing work-related musculoskeletal disorders (WMSDs) and psychological stress [35, 37–40]. In Great Britain, WMSDs have contributed to 8.8 million lost labour days in just a year from 2015 [41]. However, With the raising awareness of exposure factors in the workplace, 2018 shows a decrease of 1.9 million lost days [42] and these yearly national statistics indicate that the rate of these self-reported WMSDs are generally following a downward trend. Despite the yearly improvement, still 29% of all working days are lost due to WMSDs [42]. The growing interest to incorporate ergonomic design into the workplace is resulting in an increasingly abundant number of studies on ergonomic intervention. However, it is important to note that most of these studies do not meet the strictest criteria for scientific validity. Thus, they are not considered to be of high enough quality to draw concrete conclusions from regarding the impact ergonomic interventions have on WMSDs [43–46]. Brewer et al. [44] points out that due to the incoherent and mixed results as well as lack of high quality studies, definite conclusions cannot be made on the positive benefits. However, also agrees that ergonomic interventions when properly implemented do not result in a negative effect on WMSDs. Even with the widely held view on the lack of high quality studies, the majority of literature reviews support the opinion that ergonomic intervention can be effective in reducing WMSDs. This is even more so the case when these interventions cover multiple components to account for varying factors that can lead to WMSDs [43, 45–47]. Among these studies, it is the general consensus that interventions covering several factors as opposed to one lead to a higher probability of reducing WMSMs. Therefore, interventions should assess physical, psychosocial and individual factors [38, 43].

2.3.1 Generic Tools in Ergonomics

Over several decades of research in the field of ergonomics, a variety of tools have been developed to assist in the assessment of an interaction between a human and system. These tools can broadly be defined as either ergonomic checklists or task analysis [35]. When looking at investigating work environments, these can change over time, as well as, vary from place to place in many factors such as (i) limited workspace leading to smaller desk spaces, (ii) lack of manpower leading to increased workloads/overtime or even (iii) a lack of knowledge which may hamper performance, to name a few. This is where generic checklists ensure that regardless of how different environments are, that they still conform to certain standards and regulations. A generic checklist can only be created once there is a sound understanding of the interaction, thus to close this gap in knowledge a task analysis needs to be performed. This approach may for example involve collecting data through recording such that an interaction can be analysed in depth and its core details disseminated into a risk assessment.

2.3.2 Risk Assessment

As part of risk assessments, a number of measures exist that aim to quantify good design and working practice which assess various mechanical[3] (physical) exposure factors such as posture, range of motion, force, repetition and time [51, 52]. Risk assessments are typically designed to quickly establish the level of risk contained within an interaction. Moreover, they are meant to be easy to implement by anyone with a minimal setup cost [52]. They are not expected to provide full details with respect to ergonomics, but instead are used to indicate if a task needs to be investigated further based on the assessed level of risk. Assessments can come in the form of

[3]It is important to acknowledge that different types of risk assessments exist which take psychosocial and organisational factors into account [35, 36, 48–50], however these are out of scope and not discussed in further detail.

self-reports which are prone to user bias, or observational methods which are based on the user's actions (but can be prone to biases in the observer's interpretations of the actions) [36].

An early study on human ergonomics looks at modelling muscle fatigue using Electromyography (EMG) at varying postures and arm loads [53]. These include shoulder fatigue on varying shoulder abduction angles, vertical reaching, horizontal reaching, elbow positions and head tilt. Furthermore, different design tools and workstations are also considered which include changing the seating to include padded arm rests as well as redesigning soldering equipment. This study proposes recommendations on ideal body postures to minimise the effects of muscle fatigue arising from continuously holding certain body postures or handling varying external loads.

Since [53], a lot of practical methods have been developed for widespread use of assessing. The Ovako Working Posture Analysis System (OWAS) is one such method for whole-body posture recording and analysis [36]. It was developed mainly for work in the heavy industry by studying 32 experienced steel workers. This approach starts by observing the user, to break down a task interaction with respect to frequency and time into classified working postures. Then, each classified posture is rated and reclassified into one of four categories ranging from normal postures to ones that needs immediate attention [54]. Although it has been tested for 2 years in the location it was developed at [54], in practice the observation scheme coupled with the rules regarding frequency distribution of items can make this approach somewhat time consuming [55]. Moreover, according to Takala et al. [55] it is predominantly used in research, despite an abundance of studies which apply this method to a variety of industries including from personnel in the military, ship maintenance, chemical plants, forestry, food, automobile and farming to name a few [56]. On top of being one of the oldest observational methods [36], it also remains as one of the most popular to this day [56]. Although this approach is well established, it does have its limitations when it comes to evaluating loading or unloading actions, such as when

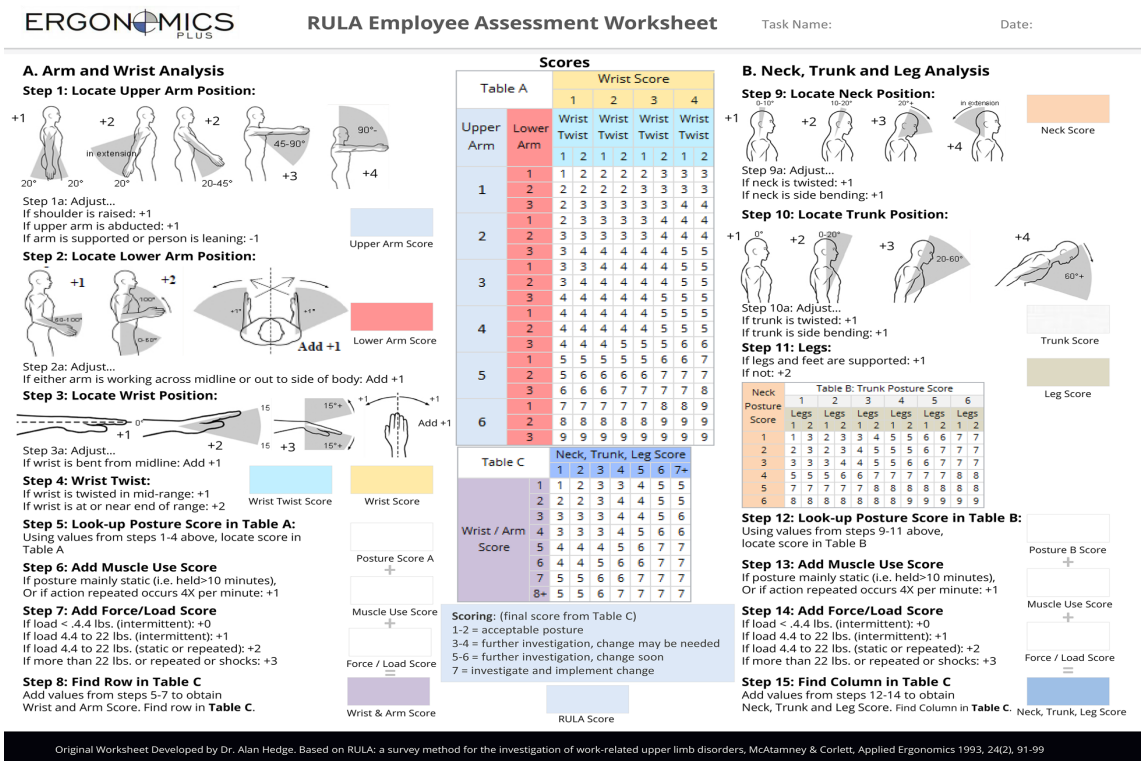


FIGURE 2.9: The Rapid Upper Limb Assessment worksheet used for evaluating the risk factor as a score. To compute the score for any static pose, follow steps 1-4 successively so that four individual scores are obtained for the upper arm, lower arm and wrist (bending and twisting), respectively. Then, enter those individual scores into the table A under 'Scores' (in the middle). To use this table, the upper arm and lower arm are used in turn to select the appropriate row, and the wrist bend score followed by the wrist twist score selects the correct column resulting in an overall score from table A. Before completing section A, steps 6-7 are performed to account for muscle fatigue and force, these points are simply added onto the previously calculated table A score giving an overall Wrist and arm score which is now kept to the side for later use. The next step is to obtain a score for the neck, trunk and leg following steps 9-11. The neck score selects the appropriate row from table B, and the trunk followed by the leg score selects the correct columns. As done in section A, the individual points from steps 13-14 accounting for muscle fatigue and force can simply be added onto the overall score from table B. Now that an overall score is obtained for both section A and B, these are used to find the correct row and column in table C, respectively, resulting in a final score which can be checked against the 'Scoring' box below the table to verify whether it is an acceptable posture or may require change.

handling cargo. It is therefore typically used in conjunction with other methods, such as Rapid Upper Limb Assessment [56].

Another widely used method [55] which focuses on work-related upper limb disorders is the Rapid Upper Limb Assessment (RULA) [57], introduced in 1993 by Mcatamney and N. Corlett [57]. It was developed together with four ergonomists and an occupational physiotherapist. This method works by scoring static poses based on OWAS [57] and applied force of individual joints of the upper limb, where an overall lower score implies that the posture is more ergonomic. Moreover, it also takes the effect of repeated actions into account. For reference, the RULA worksheet from ErgonomicsPlus© is shown in Figure 2.9 above. This assessment can only be applied to the left or right side of a subject at a time. Thus, it falls upon the observer to decide on the loading scores of the opposite side or whether it is required to perform the full assessment on the other side. This approach does not require any tools and can be used for quick assessments [57]. This quick and easy feature also means that RULA does not provide any in-depth detail and is meant more as an initial screening tool. Over the years, it has not only been applied to a practical working scenario by the authors [57] but also by many others in a wide range of industries including children using computers, use of smartphones, vehicle assembly, truck driving, and farm work to name a few [58–62].

The widely used and well established RULA has led to the creation many other assessments which use it as a basis, where some of these include the Rapid Entire Body Assessment (REBA) [63] which is quickly gaining popularity and the more recent Rapid Office Strain Assessment (ROSA) [64]. REBA was developed for a whole body evaluation by a team of ergonomists, physiotherapists, occupational therapists and nurses. The aim being to expand on the areas in which RULA and others such as OWAS does not. It includes scoring for not just static or dynamic movements but also rapid changing and unstable gravity-assisted postures. Moreover, it add angular poses for the legs and also implements a coupling score to be used in conjunction with loads. This coupling score determines the grip a user has on the load (where grip in this case does not necessarily refer to handling the load using hands but instead can refer to any part of the body). The scoring of the loads

coupling ranges from an unacceptable pose (which is defined by an awkward or unsafe grip with no handles) to a good score defined as a *well-fitting handle and uses a mid-range, power grip* [63]. This method does come with some caveats such as being time consuming, not only due to the process of recording, reviewing, posture selecting and assessment, but also as the left and right hand evaluations cannot be combined and need separate evaluations [65]. Further to this, the observer needs to decide whether to place emphasis on the loading of the muscles, posture repetition or the level of discomfort in postures. Having to prioritise one issue over another such as the 'most frequently used' over the 'worst' posture can lead to overly optimistic results, as misplaced emphasis can hide underlying issues in the task [65]. Although it is less established than traditional methods such as OWAS and RULA, REBA is already considered to be a popular choice and is seeing more and more use in the industry and services. However, it is important to note that the majority of research tend to use REBA in comparative studies [66]. ROSA is adapted more towards an office environment that incorporates computer use as well as other desk related peripherals such as the chair, monitor, telephone, keyboard and mouse [64]. It is designed to evaluate most aspects of a typical desk scenario in detail including factors such as chair height, chair depth, arm rest height, lumbar support, screen height and work surface height to name a few. Unlike methods such as RULA and REBA which are more generalised and meant as an initial screening for potential risks in many scenarios, ROSA defines specificity in the desk setup of an office and thus provides guidance on criteria which are too in-depth to otherwise be considered [67].

While researchers play the biggest role when it comes to developing observational methods, it is not uncommon for governmental bodies to initiate a study as done in 1981 by the United States' National Institute for Occupational Safety and Health (NIOSH) [68]. This led to the standard NIOSH lifting equation which focuses on assessing tasks with lifting compact loads using both hands. Moreover, it provides recommendations on how to select and train workers to reduce risk factors when

having to handle objects where weight, size, location and frequency of handling are known. The equation was later revised in 1991 to account for asymmetry, coupling and frequency in manual lifting, now referred to as revised NIOSH lifting equation [69]. Both equations are defined by three disciplines, namely (i) *biomechanical* which accounts for the spines maximum disc compression force, (ii) *physiological* accounting for maximum energy expenditure, and (iii) *psychophysical* accounting for maximum acceptable weight which is suitable for 75% of female workers and %99 of male workers. These design criteria with their cut-off value in addition to horizontal location of the object relative to the body, vertical location of the object relative to the body, the moved vertical distance of the object, asymmetrical angle, frequency of lifting, duration of lifting and finally the coupling or grip on the object give a measurement on the limits to set for the manual lifting task. This method has been used in many studies, however with mixed results regarding its robustness with respect to widespread adoption. This issue comes from there being a bias towards simpler studies due to its applicability restrictions to jobs with just lifting and lowering tasks. To this extent, several authors agree that the method needs to be refined and extended to incorporate a wider range of movements and thereby applicability such as pushing and pulling actions. Most agree that more research with larger sample sizes are required to draw definite conclusions on its efficacy [70–74].

The strain index developed by Moore and Garg [75] in 1995 assesses the risk of some distal upper extremity disorders including disorders in muscle-tendons and carpal tunnel syndrome but not all oseoarthiritis, ganglion cysts and ulnar nerve entrapment at the elbow to name a few. Similarly to NIOSH, the assessment score is derived from an equation, three of these are qualitative results collected by a job analyst, namely intensity of exertion, hand/wrist posture and speed of work, and the other three variables are the duration of exertion, exertions per minute and duration of task per day. Although preliminary tests show the methods accuracy in identifying jobs with higher risks relating to distal upper extremity disorders, the

authors do state that large scale studies are needed as well as refinement of the approach to diversify its use.

The concise exposure index (OCRA index) developed by Occhipinti [76] in 1988 is a method for assessing different types of jobs based on NIOSH. It evaluates the relationship between the recommended number of actions for upper limbs to the number performed by the user. It has the unique aspect of including force into the equation which is calculated through the maximum voluntary contaction from EMG and places a heavy emphasis on exposure from task repetition.

The quick exposure check (QEC) by Li and Buckle [77] in 1988 assesses the exposure of upper body and limbs for static and dynamic tasks. It comes as a checklist of questions answered by both the worker and observer, where answers to two coupled questions selects the rows and columns in one of several tables, producing an intermediate score and eventually a final score through all their sums. The authors do state that based on their studies using this method, improvements are required with respect to repetitive body movements [78]. Li and Buckle [77] do also point out that although the method is quick to learn and can assess a task within 10 minutes, reliability of the measurement improves with more experience of the approach. This indicates that a novice observer will not be able to assess others with utmost accuracy.

Loading on the upper body assessment (LUBA) by Kee and Karwowski [79] published in 2001, assesses postural loading on the upper body and limbs. It was developed to overcome a gap on observation methods that are composed from experimental data. This is in contrast to methods including RULA and OWAS in which the criteria of the former is built on information provided by economicists and occupational physiotherapists that make use of biomechanical and muscle function information, or in the case of OWAS using subjective steel worker data. The experimental data to construct LUBA is composed of 20 healthy male subjects, evaluated for their perceived discomfort for a given set of seated and standing static

postures at varying range of motion (ROM). The resulting method works as a checklist where scores are given to different static upper body parts, for example scores for the wrist are split into three motions, namely flexion, radial and deviation. Each of these motions are sub-categorised by different ranges of joint angles which have a corresponding score.

The Novel Ergonomic Postural Assessment method (NERPA) [80] is one of the newer approaches for upper body assessment. Sanchez-Lite et al. [80] make use of 3D modelling in CAD as well as real-time motion capture to design and study simulated models of different ergonomic workstations. With the aim to standardise posture classification by drawing from multiple observation methods reviewed in [81], Sanchez-Lite et al. [80] present their approach which is based on the RULA worksheet and compare it other methods including RULA and OCRA. While the authors do claim that their approach performs better than RULA in their tests, other studies do not agree [82]. However, due to how relatively new this approach is, a lot of larger and high quality studies are required before any claims on performance can be substantiated.

While many more established methods as well as newer approaches exist for ergonomic assessment, a full review of each and everyone of these would require a dissertation devoted specifically to ergonomics. Thus enough approaches have been discussed to have a good understanding of the current state-of-the-art with respect to this dissertation's focus. The discussed approaches are compiled in Table 2.1.

TABLE 2.1: Exposure factors assessed by different methods. Based on [51].

Reference	Technique	Posture	Load/Force	Movement frequency	Duration	Recovery	Vibration	Others*
[36]	OWAS	x	x					
[57]	RULA	x	x	x				
[63]	REBA	x	x	x				x
[64]	ROSA	x			x			x
[69]	Revised NIOSH	x	x	x	x	x		x
[75]	Strain Index	x	x	x	x			x
[76]	OCRA	x	x	x	x	x	x	x
[77]	QEC	x	x	x	x		x	x
[79]	LUBA	x						

*These include, mechanical compression, equipment, load coupling, psychosocial and individual factors

Looking at comparative studies, several that test observation methods including LUBA, REBA, NERPA, OCRA and RULA find that RULA tends to perform best or most stringently in assessing the risk factor of postures that incite muscle fatigue and discomfort, which can lead to work-related musculoskeletal disorders [82–84]. However, most studies come to the conclusion that these methods lack standardisation and cannot easily be compared as they all emphasise different aspects in a task such as either repetition or loading. This means that certain methods are better suited for different types of jobs [51, 81, 83, 85, 86]. As pointed out by Takala et al. [55], the selection and use of methods has often been based on tradition rather than on a critical evaluation, mainly due of a lack of awareness in methods outside of a researchers own realm of experience. Further to this, as mentioned throughout this chapter, something that applies to all approaches is that more large scale and high quality studies are required to assess the efficacy of these observation methods. However, while this point hasn't been reached yet, it is the case that the more traditional approaches have been validated widely and thus have more credibility than any of the newer approaches, which claim to outperform the most popular options. Although such claims could eventually be proved true, with the current lacking literature it is not possible to verify this. With respect to this dissertation, the research using human data (discussed in Chapter 5) works specifically with video from a lateral view of the demonstrator and aims to classify a large number of static postures fairly quickly in tasks which are not necessarily repetitive in a real world scenario. The emphasis primarily lies in optimising the joint configurations contained within static postures. Thus, the priority is to use a widely established approach which can also easily be adapted to remove extra evaluation criterion which are not considered in this research as done in [87] where evaluation of the neck, trunk and legs are removed. Based on all this information, RULA seems most applicable and is selected for the research conducted in this dissertation.

2.4 Singularities & Avoidance

2.4.1 Kinematic Singularities

As robots are becoming more versatile, they are increasingly performing tasks for which they were not specifically designed. As a result, there is uncertainty over the degree of redundancy (or contrastingly, overconstrainedness) in the control of their movements. When controlling a manipulator within the Cartesian space, certain inverse mappings from Cartesian space to joint space can be problematic leading to unpredictable and undesired behaviour. As a result, dealing with singularities, which are inherent in both redundant and non-redundant robotic systems, are playing an increasingly important role [88]. When a system is at a singular configuration, it loses the ability to move in a certain direction and this can occur when configurations do not have a well defined inverse kinematic solution, *i.e.,* either none or infinite solutions. In such a configuration, the system can lose a DoF such that arbitrary motions of the end-effector are not possible (see Figure 2.10). Moreover, small velocities in the operational space will generate large (infinite) velocities in the joint space

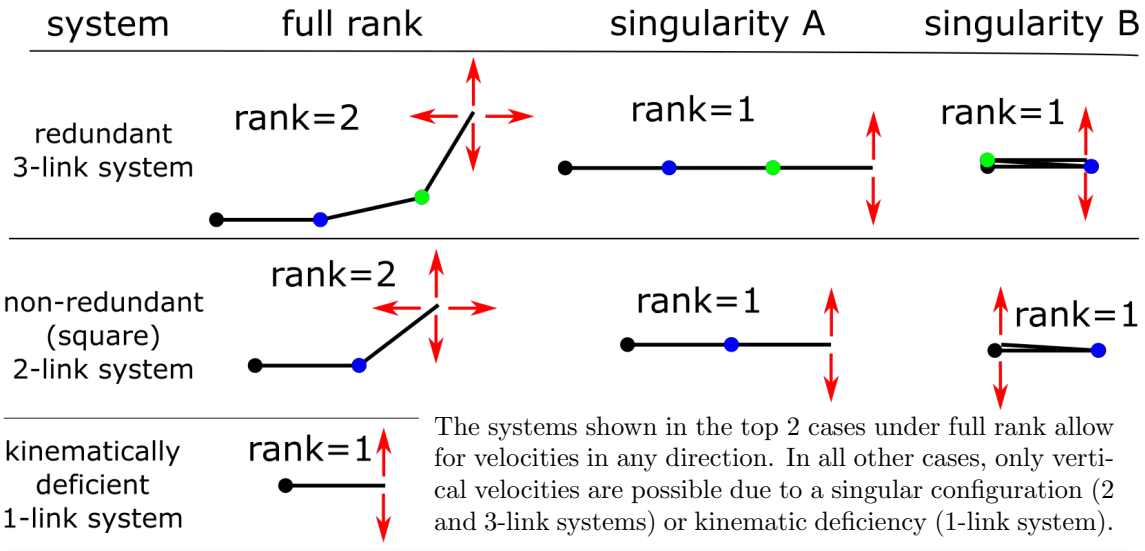


FIGURE 2.10: This shows planar systems with varying number of joints and how their possible directional velocities are affected by different configurations.

leading to unpredictable movements and deviation from the desired trajectory [89].

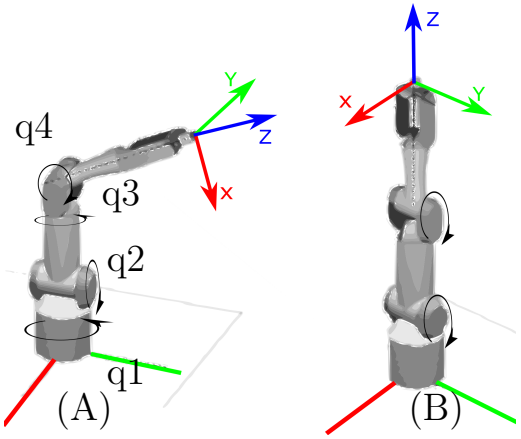


FIGURE 2.11: Singular configurations in the Mitsubishi PA-10 robot where control in the direction of the Z-axis (blue) is lost. A) on the left is an internal singularity where $q_2 = 0°$and $q_3 = \pm 90°$. B) on the right is a boundary singularity with all joints straight reaching to the furthest point within the workspace. Adapted from [90].

Kinematic singularities can broadly be broken into two types, namely (i) boundary singularities (also referred to as workspace singularities), which describe when a system is fully extended (or retracted if attempting to reach beyond its inner boundary which is closer than allowed by the joint limits). This occurs when it's expected to move outside of its reach. While no configuration of the system allows for reaching a point outside of its workspace, this singularity can be avoided by ensuring that the system only moves within its workspace (see Figure 2.11-B), (ii) internal singularities (also known as joint space singularities) occur where a system has infinite inverse kinematic solutions, such as when its axes are aligned in space (see Figure 2.11-A).

An internal singularity can be broken down further into specific types depending on where the alignment occurs such as a wrist, elbow or shoulder singularity [88]. Depending on the design of the robot, in most cases this type of singularity is avoidable by exploiting redundancy; as the same end-effector target can be reached using a different non-singular configuration (compare Figure 2.12-A and Figure 2.12-B) [91]. However, unlike boundary singularities, an

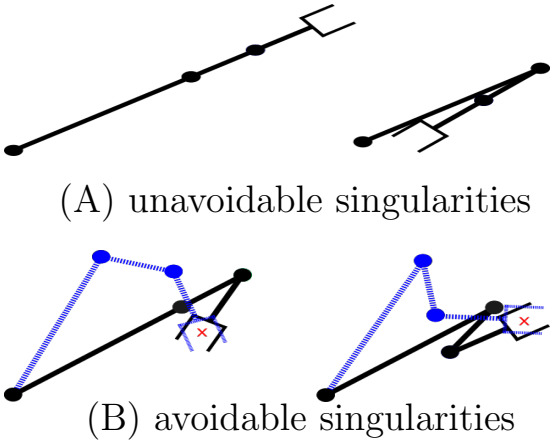


FIGURE 2.12: Example of (A) unavoidable and (B) avoidable singularities of a 3DoF planar robot. All singular configurations are shown in black, whereas sample configurations avoiding the singular pose are shown in striped blue. Adapted from [88]

internal singularity can occur anywhere in a system's workspace, resulting in small Cartesian motions requiring infinite joint velocities. In most cases a non-redundant system will have a finite set of solutions with the exact number depending on the design and amplitude of the joint limits, unless for example a target location is set outside of its reachable workspace for which there is no solution. In redundant systems or certain singular configurations, there are an infinite number of solutions. When it's possible to compute all the joint configurations for a specific end-effector position and orientation, it is said that the robot manipulator is solvable [88, 92, 93]. All non-redundant systems are considered to be solvable [88, 94, 95].

2.4.2 Jacobian Matrix & Manipulability

Analysing a system's Jacobian matrix, which describes the relationship between the instantaneous velocity of the end-effector and the joint velocities allows us to find its kinematic singular configurations [88]. As a simple approach, using the inverse of the Jacobian indicates the distance to singular configurations as it produces excessive joint velocities relative to the distance from said singular pose. In a physical system, the velocities can reach speeds too fast for the system to accurately follow [88].

Traditionally, singular issues are assessed by the so-called *manipulability* of the system, by analysing the extent to which solutions exist to the inverse problem of finding control solutions for a given set of task constraints [3]. In non-redundant systems, the singular configurations can be identified when

$$\sqrt{det(J) = 0} \tag{2.14}$$

where J is a square Jacobian. For redundant systems, *i.e.,* with a non-square Jacobian, the manipulability index can be used so that singular configurations can be

identified using the Jacobian and its transpose

$$\sqrt{det(JJ^T) = 0} \tag{2.15}$$

First introduced by Yoshikawa [3], the manipulability index works by identifying linear dependencies in the task Jacobian that may cause a singular configuration to be reached. As shown, postures moving towards a manipulability measure of 0 indicates that the system is moving towards a singular configuration. Moreover, singularities can be recognised by determining when the matrix J becomes deficient. As mentioned in Section 2.4.1, as a consequence of loosing rank, the system looses the ability to produce velocities in a certain direction of the task space. The manipulability index was initially used to devise control algorithms to avoid kinematic singularities in manipulators. It has since been used in a wide variety of contexts, such as real-time end-pose planning in walking tasks [96], grasp planning [97], and planning of human-robot interaction workspaces [20]. This measure has also been extended to account for joint limits, self-collision in redundant systems, and the need for adaptability to avoid obstacles in the workspace [98].

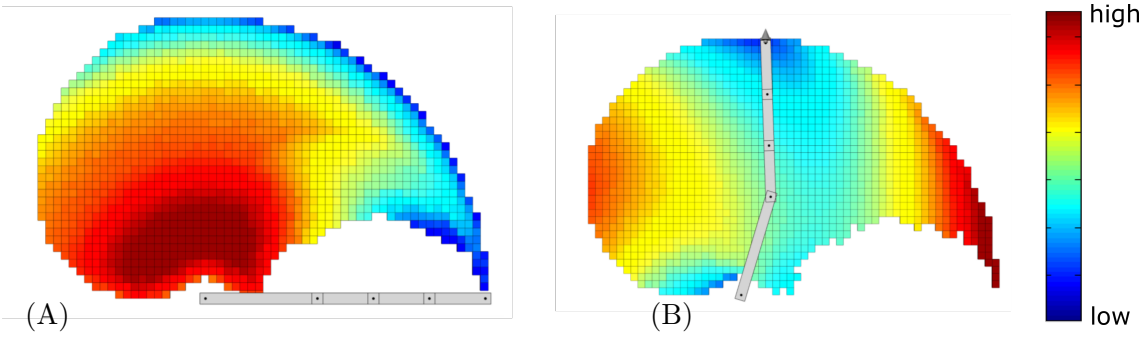


FIGURE 2.13: The maximised manipulability map for a planar 4DoF robot constructed using Yoshikawa's manipulability measure where blue is a low manipulability and red is high. (A) a standard manipulability map and (B) task-specific manipulability map constrained to an upper movement. Adapted from [98].

Knowing a system's manipulability within its workspace allows for avoidance of encountering singular configurations. As a visual representation of how analysing the

manipulability is useful, a manipulability map can be computed [99] as shown in Figure 2.13 above. Here, the map represents the system's manipulability distribution. This identifies regions in which the system has the highest manipulability, which is the preferred area to work within. In contrast, the lowest area indicates nearing a singular pose. Further to this, when the system is constrained to certain directions, its manipulability is also affected which changes the distribution of the high and low manipulability regions. As a result of this constraint, regions which previously had a few to no singular configurations can suddenly become dangerous, with a greater probability of encountering singularities subject to the constraint. Given that a system may be able to reach a particular end-effector point using multiple poses (depending on its DoF), it is possible for joint configurations of both high and low manipulability to occupy the same end-effector space. Thus, the map can be adapted to prioritise the display of either the upper or lower bounds of manipulability.

It is important to acknowledge that many implementations exist for robust singularity avoidance, which make use of the previously discussed methods as well as other alternatives, including several discussed below. Moreover, all of these have limitations when it comes to their applications or prerequisites for use. For example, in [100], which proposes an inverse kinematics algorithm based on task priority. It avoids singularities through redundancy resolution, however it does so explicitly given singular configurations as assumed knowledge *a priori*. An approach using non-linear optimisation is presented in [101] which avoids singularities by minimising a cost function. It also considers control of constrained systems. However, its main drawback is that it requires explicit knowledge of the constraints affecting the system in a given task. This may be difficult for novice uses, especially if task constraints take on complex forms (discussed in Section 4.4). In [102], a method is proposed which uses set-based tasks, *i.e., tasks with a desired interval or equivalently area of satisfaction* [103]. As each task is associated with configurations with respect to the environment, the task space target positions and orientations need to be known for each task.

With these various singularity avoidance implementations in mind, it is important to establish that Chapter 4 focuses on singularity avoidance through programming by demonstration. This assumes that most of the aforementioned prerequisites are not known, as these first need to be learnt using the constraint learning approach (discussed in Section 2.2.3). After learning, to achieve manipulability optimisation, various approaches then become applicable. However, this work considers the use of Yoshikawa's manipulability measure, as it is not only a popular choice but also lends itself well to generalisation based on the minimal input parameters required to calculate the manipulability. Most importantly, this measure works well with constraint learning, as the primary objective of learning the constraint directly results in the information needed by Yoshikawa's manipulability measure without requiring any additional steps (discussed in Section 4.5).

Chapter 3

Effects of Problem Complexity on Learning

3.1 Introduction

This chapter looks at how the increasing complexity of problems in tasks affects learning. It proposes the application of gradient descent to the, at the time, state-of-the-art constraint learning approach [23], to reduce the search space when learning constraints. This aids in mitigating issues surrounding learning in increasingly complex tasks, more specifically data dimensionality. This proposed approach makes constraint learning applicable to modern robotic systems with greater degrees of freedom (DoF) and significantly reduces the time taken to learn a constraint model. As it also has a lower demand on computation costs, this in turn expands its applicability to lower cost computers.

First, issues relating to high dimensional data are discussed. Then, the gradient descent approach and its implementation are presented. Experiments are conducted to evaluate how well it performs on data of varying dimensionalities as well as constraint types with respect to time and accuracy. It assumes no prior knowledge

regarding dimensionality of the constraints. The evaluations show how the prior state-of-the-art is limited to 4D data by the hardware used in the experiments. However, using gradient descent allows for learning on 9D data under the same hardware limitations. Furthermore, its parameters are evaluated to give an outline of how it performs when tuning for speed based on the number of starting points (elaborated in Section 3.3) against the average error.

Robot interactions with the environment tend to involve some type of constraint which may have an impeding factor to the success of a task, for example, when considering environmental constraints, we can see how the surface of a table limits movement of a hand during a wiping task (Figure 3.1) [13].

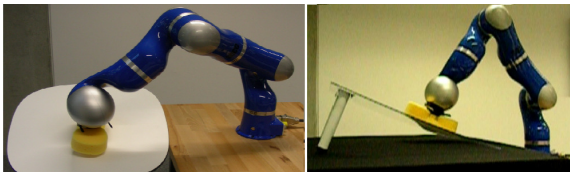


FIGURE 3.1: System restrictions on different environmental constraints [23].

On the other hand, complex interactions can also be derived from self-imposed constraints, such as pouring water from a bottle, as the hand's orientation is restricted such that the water is poured into the glass. Essentially, robotic manipulators are expected to produce a set of joint-space movements which comply with the constraint [11, 18].

One approach to learning constraints is to focus on extracting relating information from restricted motion [23], particularly where the constraint plays a critical role, such as planning a path through the manipulation of obstacles [104] or maintaining grip on a remote while switching between different buttons [13].

There are numerous approaches to learning constraints from the environment, including vision based systems [105], force/torque and tactile sensors [106] or a combination of both [107]. Recently, systems learn kinematic constraints using proprioception as done in [108, 109] and movement observation as proposed by [13].

This chapter proposes an alternative approach to [13], to learn the kinematic constraints from observed movements and, in doing so, extends the approach envisioned originally in [23]. Our method derives the null space projection of a kinematically constrained system using gradient descent. Moreover, we compare this method to [23] for learning constraints on data sets of different dimensionality to demonstrate: 1) how it can predict policies faster when dealing with higher dimensional problems, and 2) that it can learn constrained policies from data of a much higher dimensionality, with a smaller demand for computational resources.

3.2 Problem with high dimensional data

With the constant growth in data complexity and volume, machine learning plays an ever increasingly significant role when it comes to extracting important information from an abundance of unnecessary data. Increasing dimensionality in data can have detrimental effects on not just computational costs but can also lead to poor generalisation due to bellman's *curse of dimensionality* [110], which refers to how obtaining an accurate solution to a problem data set increases exponentially based on the size of the input, *i.e.,* dimensionality [111]. This problem is a prevalent issue in robotics, especially with the increasing need to have high DoF systems perform tasks in real-time, and even more so as their application expands to other fields including as support robots for humans [112]. Moreover, data is made more difficult if it is non-linear in nature making function approximation a nontrivial task [113]. Thus, it becomes vital for optimisation of learning algorithms to quickly extrapolate necessary components in data, especially for practical applications in planning real-world high dimensional tasks in unstructured changing environments.

3.3 Method: Gradient-Descent to optimise learning

This chapter considers the problem of finding a constraint from observed motion data, such as an object or obstacle in the path of a motion that restricts said movement of a system (e.g. a manipulator). We consider systems that are constrained through a set of $\mathcal{S}$-dimensional ($\mathcal{S} \leq \mathcal{Q}$) constraints of the form

$$\mathbf{A}(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) = \mathbf{0} \tag{3.1}$$

where $\mathbf{x} \in \mathbb{R}^{\mathcal{P}}$ represents state, $\mathbf{u} \in \mathbb{R}^{\mathcal{Q}}$ represents the action, t is time, and $\mathbf{A}(\mathbf{x}, t) \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ is a matrix describing the constraints. *The goal is to estimate the constraints described in* $\mathbf{A}$ *only given the observed actions* $\mathbf{u}$.

The relation between $\mathbf{A}$ and $\mathbf{u}$ can be observed by inverting Equation 3.1, resulting in

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{N}(\mathbf{x}, t)\boldsymbol{\pi}(\mathbf{x}) \tag{3.2}$$

where $\mathbf{N}(\mathbf{x}, t)$ is described in Equation 2.4.

The proposed approach requires no information regarding the dimensionality of the constraint $\mathbf{A}$ or the underlying control policy $\boldsymbol{\pi}$, it only requires the observed motion data $\mathbf{u}$ to learn the constraint matrix $\mathbf{A}$.

The key to the proposed approach is to use properties of the projection matrix $\mathbf{N}$ in order to find $\mathbf{A}$ [23]. By definition $\mathbf{u} = \mathbf{N}\boldsymbol{\pi}$, so $\mathbf{u}$ is the vector $\boldsymbol{\pi}$ projected onto the image space of $\mathbf{N}$. However, it is also the case that *the projection of* $\mathbf{u}$ *also lies in this image space*, i.e.,

$$\mathbf{N}\mathbf{u} = \mathbf{u}. \tag{3.3}$$

Based on this insight, $\mathbf{N}$ can be approximated by seeking an estimate *such that this condition holds.*

Specifically, given samples $\{\mathbf{x}_n, \mathbf{u}_n\}_{n=1}^{\mathcal{N}}$ it is proposed to form an estimate $\tilde{\mathbf{N}}$ that minimises the difference between $\tilde{\mathbf{N}}\mathbf{u}$ and the raw observations $\mathbf{u}$, i.e.,

$$E = \frac{1}{2}\sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\mathbf{N}}\mathbf{u}_n||^2. \tag{3.4}$$

Using Equation 2.4, the nth term of Equation 3.4 can be written

$$||\mathbf{u}_n - (\mathbf{I} - \tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}})\mathbf{u}_n||^2 = ||\mathbf{u}_n - \mathbf{u}_n + \tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n||^2 = ||\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n||^2. \tag{3.5}$$

where $\tilde{\mathbf{A}} \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ is an estimate of the constraint matrix $\mathbf{A}$. Expanding the norm $||\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n||^2 = \mathbf{u}_n^\top(\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}})^\top\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n$, and using the identities $(\mathbf{A}^\dagger\mathbf{A})^\top = \mathbf{A}^\dagger\mathbf{A}$ and $\mathbf{A}^\dagger\mathbf{A}\mathbf{A}^\dagger = \mathbf{A}^\dagger$, Equation 3.4 can be expressed in simplified form [23]

$$E = \frac{1}{2}\sum_{n=1}^{\mathcal{N}} \mathbf{u}_n^\top\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n. \tag{3.6}$$

Defining the data matrix $\mathbf{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_{\mathcal{N}}) \in \mathbb{R}^{\mathcal{Q}\times\mathcal{N}}$, this can be rewritten

$$E = \frac{1}{2}\mathrm{Tr}[\mathbf{U}^\top\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{U}]. \tag{3.7}$$

The constraint projection matrix can therefore be estimated by seeking an estimate $\tilde{\mathbf{A}}$ that minimises Equation 3.7. Note that, through use of the latter, *no prior knowledge of the underlying* $\boldsymbol{\pi}$*, nor of the true projection matrix* $\mathbf{N}$ *is required.*

3.3.1 Representation of A

So far, no specific assumptions have been made on the form of the estimated matrix $\tilde{\mathbf{A}}$. Following [23], an appropriate structure of this matrix is outlined, which ensures that the estimate is interpretable in terms of the constraints. Specifically, $\tilde{\mathbf{A}}$ is

assumed to be formed from a set of $\mathcal{S}$ orthonormal vectors

$$\tilde{\mathbf{A}} = \left[\boldsymbol{\alpha}_1^\top \boldsymbol{\alpha}_2^\top \cdots \boldsymbol{\alpha}_{\mathcal{S}}^\top\right]^\top \tag{3.8}$$

where $\boldsymbol{\alpha}_s = (\alpha_{s,1}, \alpha_{s,2}, ..., \alpha_{s,\mathcal{Q}})$ corresponds to the sth constraint in the observations. The latter are represented by a total of $\mathcal{P} = \mathcal{S}(2\mathcal{Q} - \mathcal{S} - 1)/2$ parameters[1] $\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_{\mathcal{P}})^\top$. Since $\tilde{\mathbf{A}}$ consists of orthonormal row vectors, $(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^\top)$ is invertible and is written as

$$\tilde{\mathbf{A}}^\dagger = \left(\frac{\boldsymbol{\alpha}_1^\top}{\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^\top} \frac{\boldsymbol{\alpha}_2^\top}{\boldsymbol{\alpha}_2\boldsymbol{\alpha}_2^\top} \cdots \frac{\boldsymbol{\alpha}_S^\top}{\boldsymbol{\alpha}_S\boldsymbol{\alpha}_S^\top}\right). \tag{3.9}$$

where $\boldsymbol{\alpha}_i = ||\boldsymbol{\alpha}_i||\hat{\boldsymbol{\alpha}}_i$. This in terms of the estimated projection matrix $\tilde{\mathbf{N}}$ is written as

$$\tilde{\mathbf{N}} = \mathbf{I} - \left(\frac{\boldsymbol{\alpha}_1^\top}{\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^\top} \frac{\boldsymbol{\alpha}_2^\top}{\boldsymbol{\alpha}_2\boldsymbol{\alpha}_2^\top} \cdots \frac{\boldsymbol{\alpha}_S^\top}{\boldsymbol{\alpha}_S\boldsymbol{\alpha}_S^\top}\right) \begin{pmatrix} \boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \\ \vdots \\ \boldsymbol{\alpha}_S \end{pmatrix} = \mathbf{I} - \sum_{s=1}^{S} \hat{\boldsymbol{\alpha}}_s^\top \hat{\boldsymbol{\alpha}}_s \tag{3.10}$$

Since $\tilde{\mathbf{A}}$ consists of orthonormal row vectors, $(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^\top)^{-1} = \mathbf{I}^{-1} = \mathbf{I}$. So the pseudoinverse here simplifies to $\tilde{\mathbf{A}}^\dagger = \tilde{\mathbf{A}}^\top$. Substituting into Equation 3.7 yields

$$E = \frac{1}{2}\mathrm{Tr}[\mathbf{U}^\top \tilde{\mathbf{A}}^\top \tilde{\mathbf{A}}\mathbf{U}]. \tag{3.11}$$

To visualise a simple case of the equations defined, we can consider a 1D constraint in a system with two degrees of freedom where $\mathbf{A} = \hat{\boldsymbol{\alpha}} \in \mathbb{R}^{1\times 2}$ and $\hat{\boldsymbol{\alpha}}$ is a unit vector representation of $\boldsymbol{\alpha}$ (Fig. 3.2) [23]. Figure 3.2 shows how the $\boldsymbol{\theta}$ corresponds to different estimates of the

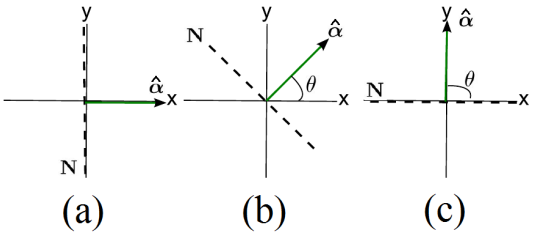


FIGURE 3.2: Three 1D unit vector constraints $\hat{\boldsymbol{\alpha}}$ where (a) $\theta = 0^\circ$, (b) $\theta = 45^\circ$, and (c) $\theta = 90^\circ$ (reproduced from [23]).

[1]Since each $\boldsymbol{\alpha}_s$ is orthonormal to all preceding vectors $\boldsymbol{\alpha}_1, \cdots \boldsymbol{\alpha}_{s-1}$, the number of parameters required to describe the sth vector is $\mathcal{Q} - s$. The total number of parameters required to describe $\tilde{\mathbf{A}}$ then is $\mathcal{Q} - 1 + \mathcal{Q} - 2 + \cdots + \mathcal{Q} - \mathcal{S} = \mathcal{S}\mathcal{Q} - \sum_{s=1}^{\mathcal{S}} s = \mathcal{S}(2\mathcal{Q} - \mathcal{S} - 1)/2$.

constraint $\hat{\boldsymbol{\alpha}}$ as well as its associated null space $\mathbf{N}$. In this example, $\hat{\boldsymbol{\alpha}}$ is simply a unit vector with the form $\hat{\boldsymbol{\alpha}} = (\cos\theta, \sin\theta)$ [23].

3.3.2 Gradient Descent

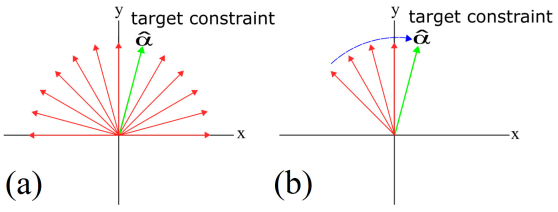


FIGURE 3.3: The search space for (a) brute force and (b) gradient descent.

Although the brute force method and gradient descent approach work on the same insights, the main deviation and thereby main contribution lies in how the search for every $\tilde{\boldsymbol{\alpha}}$ is conducted as demonstrated in Figure 3.3. The former method Figure 3.3-A performs an exhaustive search by generating a set of $\mathcal{L}$ sample $\boldsymbol{\theta}$s and finding the one with the lowest error (*i.e.,* the target constraint) from all possible solutions. While, the latter approach finds the optimal $\boldsymbol{\theta}$ by first initialising a random $\boldsymbol{\theta}$ which lies between 0 and π (thus covering the entire search space, including between π and 2π which are identical [23]), and iteratively moving towards the negative of the gradient in fixed step sizes ω.

This iterative approach to find the optimal $\boldsymbol{\theta}$ can be expressed as:

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \omega \boldsymbol{\nabla}_{\boldsymbol{\theta}} E. \tag{3.12}$$

Applying the chain rule

$$\boldsymbol{\nabla}_{\boldsymbol{\theta}} E = \frac{\partial E}{\partial \tilde{\mathbf{A}}} \frac{\partial \tilde{\mathbf{A}}}{\partial \boldsymbol{\theta}} = \tilde{\mathbf{A}}\mathbf{U}\mathbf{U}^{\top} \frac{\partial \tilde{\mathbf{A}}}{\partial \boldsymbol{\theta}} \tag{3.13}$$

where the identity $\frac{\partial}{\partial \mathbf{A}} \mathrm{Tr}[\mathbf{B}^{\top}\mathbf{A}^{\top}\mathbf{A}\mathbf{B}] = 2\mathbf{A}\mathbf{B}\mathbf{B}^{\top}$ is used.

One important aspect of this approach is having k number of randomly distributed starting $\boldsymbol{\theta}$s, which can run in parallel with a condition to stop the search if one

of the $\boldsymbol{\theta}$s reaches the constraint sooner, saving both time and memory demand through fewer overall iterations. Multiple $\boldsymbol{\theta}$s reduce the chance of getting stuck in a minimum, differing from the desired constraint. It is also necessary to set a limit to the number of iterations j. Otherwise the time required to reach the minimum cannot be precisely predicted, resulting in vastly varying running times. In [23], the learning procedure using the brute force method is described.[2]

3.4 Experiments on Simulated Data

This experiment aims to assess whether the gradient descent method learns null space projections from constrained data faster than the brute force approach in [23], and identify the strengths and weaknesses of the method by evaluating its performance under varying conditions.

Following [23], a linear and non-linear ground truth null space policy type ($\boldsymbol{\pi}$) are considered. These policies are designed to simulate real world manipulators of varying degrees of freedom, and are tested on problems where $Q = 1, \ldots, 9$:

1. a linear policy: $\boldsymbol{\pi}(\mathbf{x}) = -\boldsymbol{L}(\mathbf{x} - \mathbf{x}^*)$ where $\boldsymbol{L}$ is a $Q \times Q$ positive definite gain matrix and $\mathbf{x}^*$ is the target state.
2. a sinusoidal policy:

$$\boldsymbol{\pi}_i(\mathbf{x}) = \begin{cases} \pi i \sin x_1 ... \sin x_{Q-1} \text{ for } i \text{ odd} \\ \pi i \cos x_1 ... \cos x_{Q-1} \text{ for } i \text{ even.} \end{cases}$$

To evaluate the performance of the gradient descent approach, the following parameters are used. A fixed step ω size, the number of iterations as $j = 100$ and $k = 20$ initial $\boldsymbol{\theta}$s drawn uniform randomly are used. Following [23], the training and testing

[2] An implementation of the algorithm described in this section can be downloaded from github.com/jeevanmanavalan/constraint-learning

data sets each contain 50 data points which are drawn uniform-randomly across the space $(\mathbf{x})_i \sim \mathcal{U}(-2, 2), i \in \{1, 2\}$ and are subject to a 1D constraint $\boldsymbol{A} = \hat{\boldsymbol{\alpha}} \in \mathbb{R}^{1\times Q}$, in the direction of the unit vector $\hat{\boldsymbol{\alpha}}$. The constraint is generated uniform-randomly, $\boldsymbol{\theta} \sim \mathcal{U}(0, \pi)$ *rad*.

The experiment is repeated 50 times and the average Projected Policy Error (PPE) (defined in Appendix A.1) and time taken to learn the constraint are evaluated. The experiment is repeated with the brute force method on the same data. In addition to this, a further experiment is conducted to test how the method performs under different values of k.

3.4.1 Linear and Non-linear Policy

Figure 3.4 shows the average Projected Policy Error (PPE) (defined in Appendix A.1) obtained for brute force and gradient descent. These methods are used to learn a 1D constraint from data ranging between 2-9 dimensions. The solid (blue)

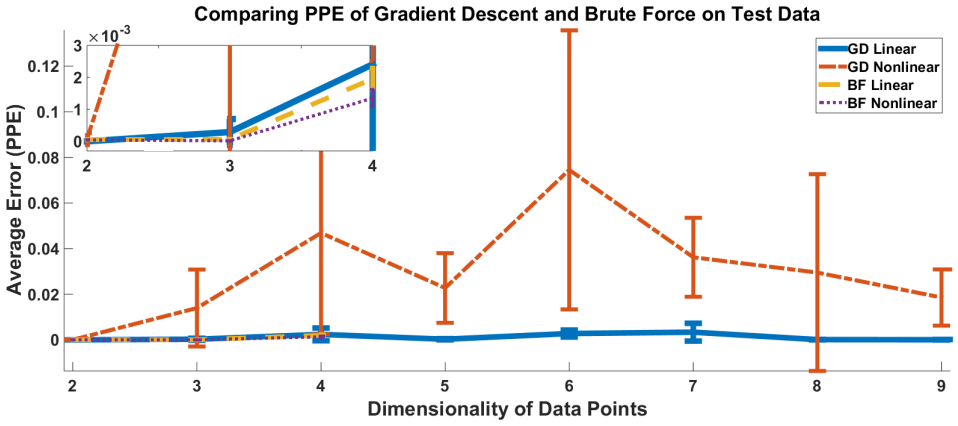


FIGURE 3.4: PPE (mean±std) of Gradient Descent and Brute Force for 2-9 Dimensions over 50 trials. Inset showing same data for 2-4 dimensions.

and dashed (yellow) lines in Figure 3.4 show good predictions from linear data with both gradient descent and brute force respectively (with errors around $\sim 10^{-3}$ shown clearly in the inset within the Figure). The main result highlighted from this plot is that gradient descent can learn constraints of up to 9D data, while brute force is limited to 4D data, since generating a search space for 5D data requires over 30GB of

memory using Matlab. The significant difference in memory resource requirements demonstrates gradient descent's superiority with linear control policies. Although this method performs with a lower accuracy when dealing with non-linear data, the errors ranging between 0.01 and 0.08 remain below 0.1%. Both the rate and standard deviation of errors in gradient descent-based trials, relating to problems with local minimums and increasing search space sizes, can be addressed by increasing j or k. There is no mentionable decline in the brute force's performance of either control policy up to 4D data.[3]

3.4.2 Time & Accuracy

Another important factor for practical applications is the time these methods take when handling data of varying dimensionality. Figure 3.5 shows the average time taken for learning the constraint using the aforementioned methods. The specification of the computer used is an Intel Core i5-4690 CPU with 4 CPU's running at 3.5GHz and 8GB of memory. No background programs were running during testing. In relation to 2D and 3D data, brute force learns fastest, requiring on average a fraction of a second. However, with 4D data, brute force takes 9 times longer than its competitor due to a large search space. By contrast, gradient descent with 9D data and using $j = 100$ takes maximum 5 seconds under restricted conditions. As explained in Section 3.4.1, good

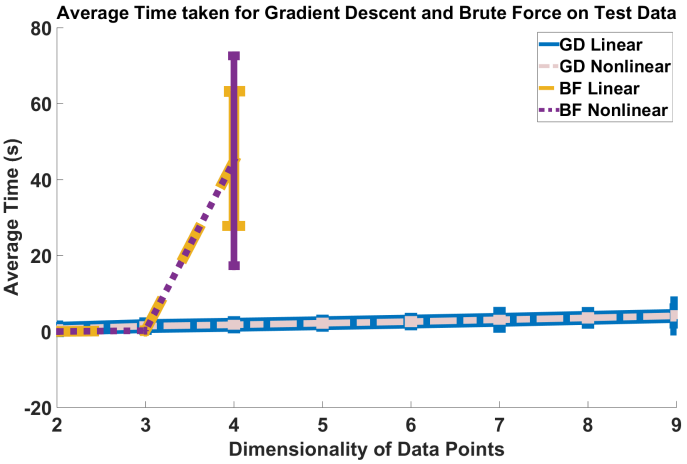


FIGURE 3.5: Time taken (mean±std) for gradient descent and brute force over 50 trials with std scaled 30 times.

[3]The data used in these evaluations can be downloaded from https://doi.org/10.6084/m9.figshare.4645456

predictions can be made especially in the case of handling a linear control policy, due to not having to deal with getting stuck in local minimums.

3.4.3 Testing varying number of starting points for Gradient Descent

At this point, the efficacy of gradient descent over brute force with respect to higher dimensional data and computation time has been demonstrated. It is also of interest to see the extent to which varying parameters of gradient descent can affect learning performance. This experiment demonstrates how learning is affected by different numbers of starting estimates. Figure 3.6 shows the average PPE and its standard deviation for a varying number of starting points on 5D data using a linear control policy. As shown, a greater number of starting estimates not only reduces the overall error but also the standard deviation, which results in more consistently similar error

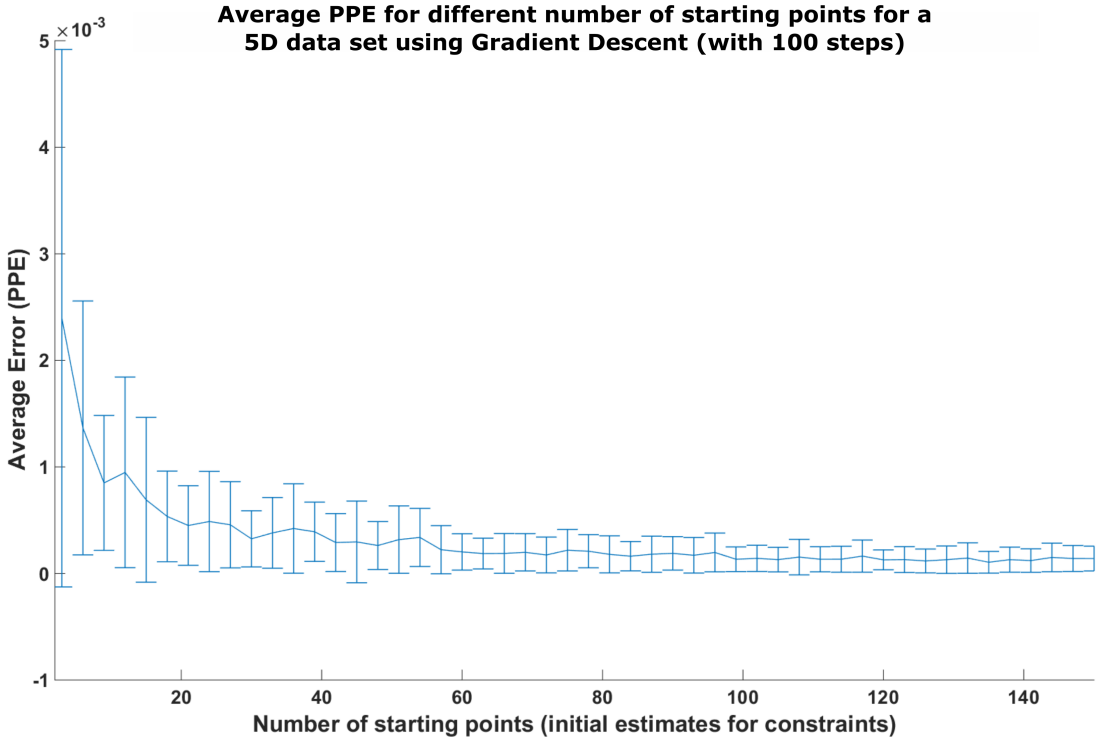


FIGURE 3.6: Effect on PPE for varying number of starting points over 50 trials.

rates. It is also important to note that even with a few starting points, we usually can predict the constraint with errors below $\sim 10^{-3}$.

3.5 Conclusion

This chapter looked at the performance of the gradient descent-based learning method. It is shown how this method can be used to estimate the null space projection matrix of a kinematically constrained system; in the absence of any prior knowledge regarding the underlying control policy. In the experiments, the proposed method is compared to the, at the time, state-of-the art method in [13] which uses brute force to estimate the constraints. The findings reveal that this novel method is significantly faster than brute force when handling 4D data or higher using a linear policy. There is a comparable yet insignificant degradation in the gradient descent's performance on non-linear data, however it can handle higher dimensions of up to and including 9D data. This significance in handling higher dimensional data is evident as brute force is limited to only 4D data. This is due to technical limitations stemming from a high demand on computer memory (see Section 3.4.1), which is not satisfied by the 8GB RAM of the computer used in the tests (see Section 3.4.2).

Learning by demonstration comes with a lot of benefits, especially with respect to simplified use by naive users (see Section 2.2). This also means that due to a user's unfamiliarity with a system, demonstrations may be performed in a way which is easier for the demonstrator rather than taking the system's structure into account. To tackle this issue revolving around human to system adaptability, with the help of the new gradient descent-based method and its applicability to higher dimensional data, the next chapter looks at how to learn singularity avoidance from demonstrations. It proposes a new approach which makes use of Yoshikawa's manipulability analysis (see Section 2.4 for background on manipulability analysis).

Chapter 4

Learning Singularity Avoidance

4.1 Introduction

This chapter looks at the issues revolving around transferring kinaesthetic demonstrations to robotic systems, focusing on how demonstration data can be used to optimise the taught systems redundancy. Learning by demonstration plays a significant role for use by naive users, as it can be assumed that the typical user has little to no understanding of the system. This gap in knowledge should be taken into account when teaching systems through demonstrations. Therefore, in this chapter a new method is proposed where guided demonstrations are used to learn task-oriented behaviour and also to resolve the system's redundancy. Through this approach, the learnt constrained manipulability, obtained from demonstration data, can be used by taught systems in their null space control policy, for manipulability maximisation without interfering with the task-oriented behaviour. It works without explicit knowledge of the dimensionality of constraints nor underlying control policies. This work considers the control of systems subject to uncertain constraints from a set of candidate constraints. This not only enables greater complexity in tasks but also allows for more accurate learning with significantly less data. This makes it a more

viable option for practical applications in the real world, such as when teaching a 7 degrees of freedom (DoF) robot in a drawer closing task with a linear constraint (see Section 4.6.3) as well as a reaching task with a non-linear constraint (see Section 4.8.3), tested in the joint space and end-effector space, respectively. Moreover, in these tests, manipulability maximisation is done locally, however the resource to implement global approximation is provided (see Section 4.5). Using this approach, systems can navigate their DoF during a task such that they avoid singularities through manipulability maximisation.

In recent years, there has been a booming shift from the development of specialised factory robots to versatile autonomous ones that are targeted at non-expert users. However, systems are increasingly performing tasks for which they were not specifically designed and there is uncertainty over the degree of redundancy (or contrastingly, overconstrainedness) in the control of their movements. Consequently, increasing importance is placed on improving control techniques to reduce such uncertainty, which otherwise may inadvertently affect the task at hand.

Traditionally, those issues are assessed by the so-called *manipulability* of the system, by analysing the extent to which solutions exist to the inverse problem of finding control solutions for a given set of task constraints. First introduced by Yoshikawa [3], the manipulability index works by identifying linear dependencies in the task Jacobian that may cause a singular configuration to be reached.

Initially used to devise control algorithms to avoid kinematic singularities in manipulators, it has since been used in a wide variety of contexts, such as real-time end-pose planning in walking tasks [96], grasp planning [97], and planning of human-robot interaction work spaces [20]. This measure has also been extended to account for joint limits, self-collision in redundant systems, and the need for adaptability to avoid obstacles in the work space [98].

In this chapter a data-driven approach is provided in which the constrained manipulability is *learnt from user demonstrations* from a set of candidate constraints. This is beneficial to non-expert users without explicit knowledge of how task constraints may affect a system and how to manoeuvre the system around them, as the approach autonomously optimises a system's control to avoid instabilities caused by singularities when carrying out a task. The proposed approach uses constraint consistent learning (CCL) [13, 18, 31] to, first, learn the task constraint and, second, optimise the manipulability derived from the learnt constraint matrix within the null space of the primary task constraint. It thereby, avoids singularity by maximising the *learnt manipulability* throughout the motion of the constrained system. Results of estimating the manipulability subject to task constraints show errors less than 10^{-5} in 3DoF simulated systems and less than 10^{-2} using a 7DoF real world robotic system for singularities occurring in the joint space.[1] Further experiments are performed to evaluate the performance of the proposed method on singularities in the end-effector space for a 2D simulation, 3D simulation and real world setting.

4.2 Problem Definition

This work considers the control of systems subject to uncertain constraints from a set of candidate constraints due to the complexity and/or naivety of non-expert users, and the need to prioritise joint configurations, which lead to greater degrees of freedom to flexibly perform demonstrated tasks.

4.2.1 Task Prioritised Constraints

Formally, a system of $\mathcal{S}$-dimensional (self-imposed or environmental) constraints can be defined as done in Equations 2.2-2.4 in Section 2.2.2. This does not only apply to

[1] It is important to understand that variables of the learning algorithm can be adjusted trading speed for accuracy depending on the intended use (see Section 3.4).

kinematics, but also to redundant actuation [22], and redundancy in dynamics [19].

Commonly, in the context of programming by demonstration by non-expert users, $\mathbf{A}, \mathbf{N}, \mathbf{b}$ and $\boldsymbol{\pi}$ are not explicitly known. Instead, the controller must be derived from data. In this chapter, it is assumed that data is given as pairs of $\mathcal{N}$ observed states $\mathbf{x}_n$ and actions $\mathbf{u}_n$.

What non-expert users may not be aware of is that $\mathbf{w}$ controls the additional free non-task related degrees of freedom of the system. It can be thought of as a lower priority task which does not conflict with the goal defined in the task space component. The benefit of having a null space component is evident in tasks which have multiple solutions. For example, a reaching task where multiple paths to a goal may be available, some paths may drive a system closer to its joint limits or singular configurations which can lead to an increased risk of getting stuck or cause turbulent movement in face of perturbations or the imposition of additional constraints.

4.2.2 Example: Opening Drawers

As a real world example, consider placing the robot in an environment designed for use by people such as an office. A robot trained to assist staff with, for example, collecting documents needs to be able to perform tasks such as reaching for and opening filing cabinets. In this case, the problem of teaching the robot to open and close drawers can be solved through programming by demonstration (as shown in Figure 4.1 below). Programming by demonstration is suitable because office staff can decide to retrain systems when it's required in another office or if the room layout changes without having any expert knowledge of the system and the need to call in a specialist *i.e.,* the user does not need to know about. This means that the approach is applicable for users without any knowledge regarding how the rows of the Jacobian matrix affects the task during demonstration (this representation is discussed in Section 4.4 and a real world example is given in Section 4.6.3). The task

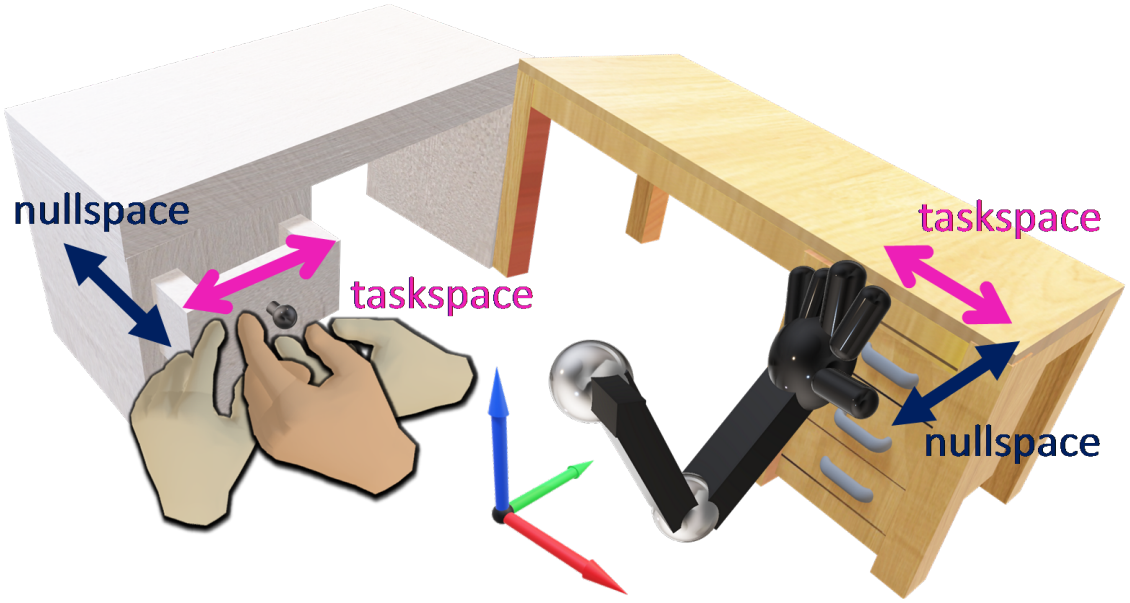


FIGURE 4.1: Demonstrating the task where the task space component shows reaching the drawer in various poses and a null space component of opening it subject to its linear constraint.

policy is to have the robot end-effector reach the drawer, after which the constraint matrix $\mathbf{A}$ is determined by a linear constraint following the direction of the drawer opening.

Where the task is kinaesthetically demonstrated, the user might manually guide a robot facing a filing cabinet, from holding onto a random point on the handle to opening the drawer towards the robot, and repeating this action multiple times using different starting points. In this scenario, the *constraint* might be learnt in a straightforward manner using one of several constraint learning methods (see Section 4.4).

Unless explicitly instructed, however, the user might not take specific care of how the motion appears when performing those demonstrations. For example, the user might choose to move towards and grab the drawer from the nearest point or in a pose/grip which allows demonstrations in the most comfortable manner for the user. It is unlikely that an average user will know to avoid unstable or singular

configurations—however, these may occur in various task-dependent situations. If, for example, a system placed in front of a drawer starts with all the joints fully extended, and a novice user then directs it to open the drawer by moving the arm directly back towards itself without resolving redundancy in the joint space, the system may propel itself in unpredictable directions due to the singularity. Moreover, systems that use an *ad hoc* way of dealing with singularities, such as when using Matlab's pseudoinverse function which replaces singular values with zero are not satisfactory, as this prevents any movement of the system and thus comes at the expense of completing the task. A better way to do this is by maximising the system's manipulability.

4.2.3 Task Manipulability and Programming by Demonstration

To avoid these problems and reduce uncertainty of encountering unstable or singular configurations (Section 4.2.2), traditionally, Yoshikawa's manipulability index is used, whereby the null space degrees of freedom are used to maximise the distance from singular points during the execution of the primary task.

The manipulability is a measure of a system's ability to position and orientate its end-effector. In order to help with the designing and control of systems, Yoshikawa developed the manipulability index [3]. It works by finding the linear dependencies in the task Jacobian which could result in reaching a singular configuration. Thus the following measure is used to assess manipulability

$$\mu(\mathbf{x}) = \sqrt{\det\left(\mathbf{A}(\mathbf{x})\mathbf{A}(\mathbf{x})^{\top}\right)}. \tag{4.1}$$

Note that, *computation of the manipulability presupposes the availability of* $\mathbf{A}$ *in analytical form*—however, as noted in Section 4.2.1, this is usually unavailable in

the context of programming by demonstration where the primary task and associated constraints are *implicit in the demonstrations*.

As $\mu \rightarrow 0$, the manipulability index indicates that the system is approaching a singular pose. The upper limit of μ depends on the system itself and can only be provided once the entire work space is assessed (however the proposed method does not require this as it compares its manipulability locally). One of the applications of μ lies in its use as a cost function to replace $\boldsymbol{\pi}$ in Equation 2.3 which results in the secondary task moving the system towards the goal using a path which favours higher manipulability (see Section 4.5).

To obtain μ, such that it can be used in such a manner, it is proposed to *form an estimate of the constraints in a task and derive an estimate of Equation 4.1*. The approach uses demonstration data such that it is applicable even for non-experts with no formal knowledge of the constraints involved in a task. In doing so, it allows for the robot to more accurately manoeuvre towards targets while minimising the risk of crossing over singular points and avoiding potentially unstable, unpredictable behaviour.

4.3 Method

In this chapter, the proposed approach forms an estimate of the task space constraint matrix $\mathbf{A}$ to be used in $\boldsymbol{\pi}$ to manipulate systems away from singular points while performing a task. This minimises the risk of encountering singularities which lead to unpredictable behaviour.

4.3.1 Data Collection

The proposed method works on data given as $\mathcal{N}$ tuples of observed states $\mathbf{x}_n$, actions $\mathbf{u}_n$ and control policies $\boldsymbol{\pi}_n$ collected though kinaesthetic demonstrations of the primary task. Assumptions on the data include that (i) observations are in the form presented in Equation 2.3, (ii) $\mathbf{b}$ from the task space varies throughout all observations, and (iii) $\mathbf{A}$, $\mathbf{N}$, and $\mathbf{b}$ are not explicitly known for any given observation.

4.3.2 Separating the task and null space components

Given the demonstration data $\{\mathbf{x}_n, \mathbf{u}_n\}_{n=1}^{\mathcal{N}}$, the first step is to separate the task and null space components. For this, the approach first proposed in [29] can be used.

As shown there, the first and second terms of Equation 2.3 can be separated by seeking an estimate $\tilde{\mathbf{w}}$ that minimises

$$E[\tilde{\mathbf{w}}] = ||\tilde{\mathbf{P}}_n \mathbf{u}_n - \tilde{\mathbf{w}}_n||^2 \tag{4.2}$$

where $\tilde{\mathbf{w}}_n := \tilde{\mathbf{w}}(\mathbf{x}_n)$ and $\tilde{\mathbf{P}}_n := \tilde{\mathbf{w}}_n \tilde{\mathbf{w}}_n{}^\top / ||\tilde{\mathbf{w}}_n||^2$. This works on the principal that, there exists a projection $\mathbf{P}$ for which $\mathbf{P}\mathbf{u} = \mathbf{P}(\mathbf{v} + \mathbf{w}) = \mathbf{w}$.

Similarly, $\tilde{\mathbf{v}}$ is required as it functions as the primary task controller for the system and can be extracted by subtracting the newly estimated $\tilde{\mathbf{w}}$ from $\mathbf{u}$, *i.e.,* $\tilde{\mathbf{v}} = \mathbf{u} - \tilde{\mathbf{w}}$.

4.4 Representation & Learning of the Constraint A

At this point, the original demonstrated actions $\mathbf{u}$ are separated into the task and null space parts. Based on the latter, the goal now is to compose an estimate of $\mathbf{A}$ that can be used in assessing manipulability. Constraints imposed on motion in the task space can refer to translational and orientational coordinates in the end-effector or joint space depending on the task at hand. An important criterion for using the manipulability for control is that constraints are *state-dependent* (explained in Figure 2.7) [23]. Otherwise, if $\mathbf{A}$ is constant across the state space [13], every state has the same manipulability index and the singularity avoidance controller has no role.

Taking this into consideration, a suitable representation of the constraint matrix is [13]

$$\mathbf{A}(\mathbf{x}) = \mathbf{\Lambda}\mathbf{\Phi}(\mathbf{x}) \tag{4.3}$$

where $\mathbf{\Lambda} \in \mathbb{R}^{\mathcal{S}\times\mathcal{P}}$ is an unknown selection matrix (to be estimated in the learning) and $\mathbf{\Phi}(\mathbf{x}) \in \mathbb{R}^{\mathcal{P}\times\mathcal{Q}}$ is a feature matrix assumed known *a priori*. The rows of the latter contain candidate constraints that can be predefined where there is prior knowledge of potential constraints affecting the system, or can take generic forms such as a series of polynomials. For example, one may choose $\mathbf{\Phi}(\mathbf{x}) = \mathbf{J}(\mathbf{x})$, the Jacobian of the manipulator, where $\mathbf{A}(\mathbf{x}) = \mathbf{\Lambda}\mathbf{J}(\mathbf{x})$ encodes constraints on the motion of specific degrees of freedom in the end-effector space.

Figure 4.2 below shows the manipulability map constructed through μ for a 2-link planar robot. The state refers to the joint angle position *i.e.*, $\mathbf{x} := \mathbf{q} \in \mathbb{R}^2$. The task space is described by the end-effector coordinates $\mathbf{r} = (r_x, r_y, r_\theta)^\top$ referring to the positions and orientation, respectively. As can be seen, singular positions in a simple unconstrained system occurs when the second joint is parallel to the first

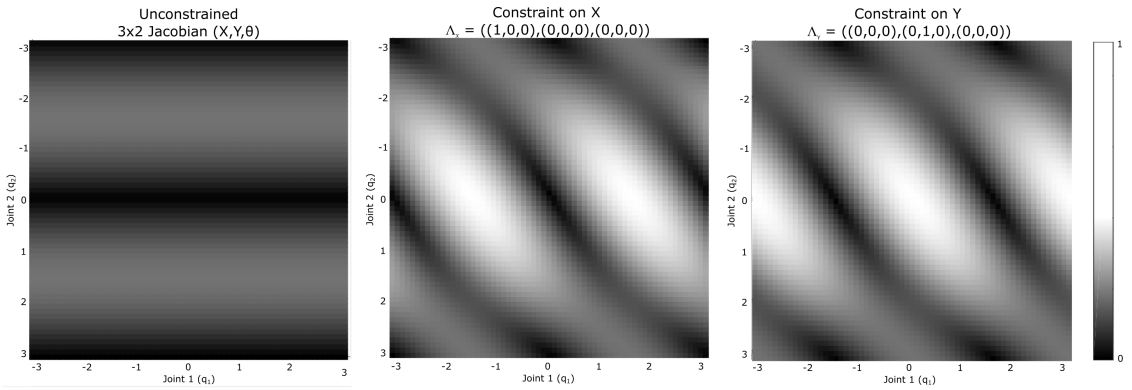


FIGURE 4.2: Manipulability Map for a system with a 3×2 Jacobian. Left is the unconstrained system. In the middle, the system is subject to a constraint on the x-axis. On the right, the system is constrained to motion on the y-axis. The manipulability is scaled between 0 and 1 with the former representing singular regions.

which is when it looses one DoF. However, when a system is constrained, the regions of high and low manipulability change accordingly. Understanding this change in a system's manipulability can affect whether a system is able to successfully complete a task, for example when the system is fully extended, this pose usually represents a singularity subject to the system being unconstrained or bound to the x-axis. On the other hand, this same pose becomes a region of high manipulability if the constraint is applied to the y-axis.[2]

In a more general form, the feature matrix $\mathbf{\Phi}$ can represent a wider variety of constraints instead of the standard Jacobian manipulator, which can make the manipulability more difficult to navigate through. Figure 4.3 below depicts the manipulability map under two non-linear constraints, namely the circular and parabolic constraints in a 2D end-effector space. As shown, the optimal control strategy changes drastically based on the constraint it is placed under. This means that for the same task, various constraints can lead to different areas of the workspace

[2]A visual representation of the system subject to a constraint under θ is not included as it is not state-dependent and would have in the same value for every possible pose resulting in a single coloured map. In this case, the resulting A matrix consisting of the bottom row of the Jacobian (selecting motion data to control θ) is always a state-independent (unchanging) scalar value as this is obtained by adding the derivative of every angle for each joint.

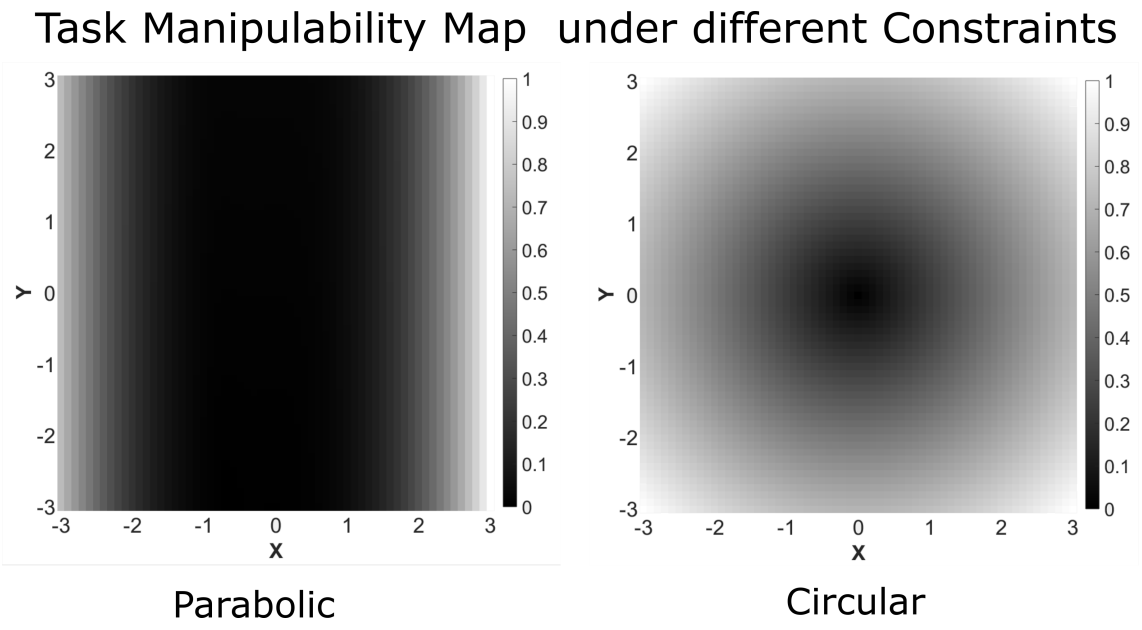


FIGURE 4.3: 2D Task Manipulability Map of different constraints. For the parabolic constraint $\mathbf{\Phi} = [2\mathbf{x}_1, -1]$. For the circular constraint $\mathbf{\Phi} = [2\mathbf{x}_1, 2\mathbf{x}_2]$. The manipulability index is scaled to range from 0 (singularity) to 1.

containing singular points. This constraint dependant variation in manipulability stresses the importance of needing to know how to avoid jeopardising the task, otherwise potentially putting people at harm through unpredictable behaviour from when the system crosses these singular points.[3]

Depending on the assumptions made on the representation of $\mathbf{A}$, one of several learning methods could potentially be used to form the estimate of the selection matrix $\tilde{\mathbf{\Lambda}}$ [13, 23, 30]. Of these, this chapter picks [23] as it requires relatively few parameters and little data to perform robustly [13]. The estimate is formed by minimising

$$E[\tilde{\mathbf{A}}] = \sum_{n=1}^{\mathcal{N}} \tilde{\mathbf{w}}_n^{\top} (\tilde{\mathbf{\Lambda}} \mathbf{\Phi}_n)^{\dagger} \tilde{\mathbf{\Lambda}} \mathbf{\Phi}_n \tilde{\mathbf{w}}_n \tag{4.4}$$

where $\mathbf{\Phi}_n := \mathbf{\Phi}(\mathbf{x}_n)$. This results in the estimate $\tilde{\mathbf{A}}(\mathbf{x}) = \tilde{\mathbf{\Lambda}} \mathbf{\Phi}(\mathbf{x})$.

[3] A visual representation of a linear constraint is not included as it is not state dependent and would have in the same value for every possible pose resulting in a single coloured map.

4.5 Estimating the Singularity Avoidance Policy

Using the estimated constraint matrix $\tilde{\mathbf{A}}$, it is now possible to form the *estimate of the system manipulability* for any configuration within the support of the data. The latter is given by substitution of $\mathbf{A}$ in Yoshikawa's manipulability index.

$$\tilde{\mu}(\mathbf{x}) = \sqrt{\det\left(\tilde{\mathbf{A}}(\mathbf{x})\tilde{\mathbf{A}}(\mathbf{x})^{\top}\right)}. \tag{4.5}$$

From this constrained manipulability map, states for particular end-effector poses can be selected based on $\tilde{\mu}$ using $\boldsymbol{\pi}$ to update the joint angles. When used as a cost function, this information can provide the direction in which the system should move. Following this increases its manipulability and maximise the distance from singular points, thereby reducing the risk of unpredictable behaviour. The simplest such approach is to use gradient ascent by replacing $\boldsymbol{\pi}$ in Equation 2.3 with

$$\boldsymbol{\pi}_{\tilde{\mu}}(\mathbf{x}) = \nabla_{\mathbf{x}}\tilde{\mu}. \tag{4.6}$$

This achieves manipulability maximisation locally through comparisons of neighbouring states. Alternatively, if the task space trajectory is predictable, $\tilde{\mu}$ can be used in combination with global approximation in the null space (see, *e.g.*, [114]).

Figure 4.4 below shows a brief overview of the major steps involved in the proposed approach.

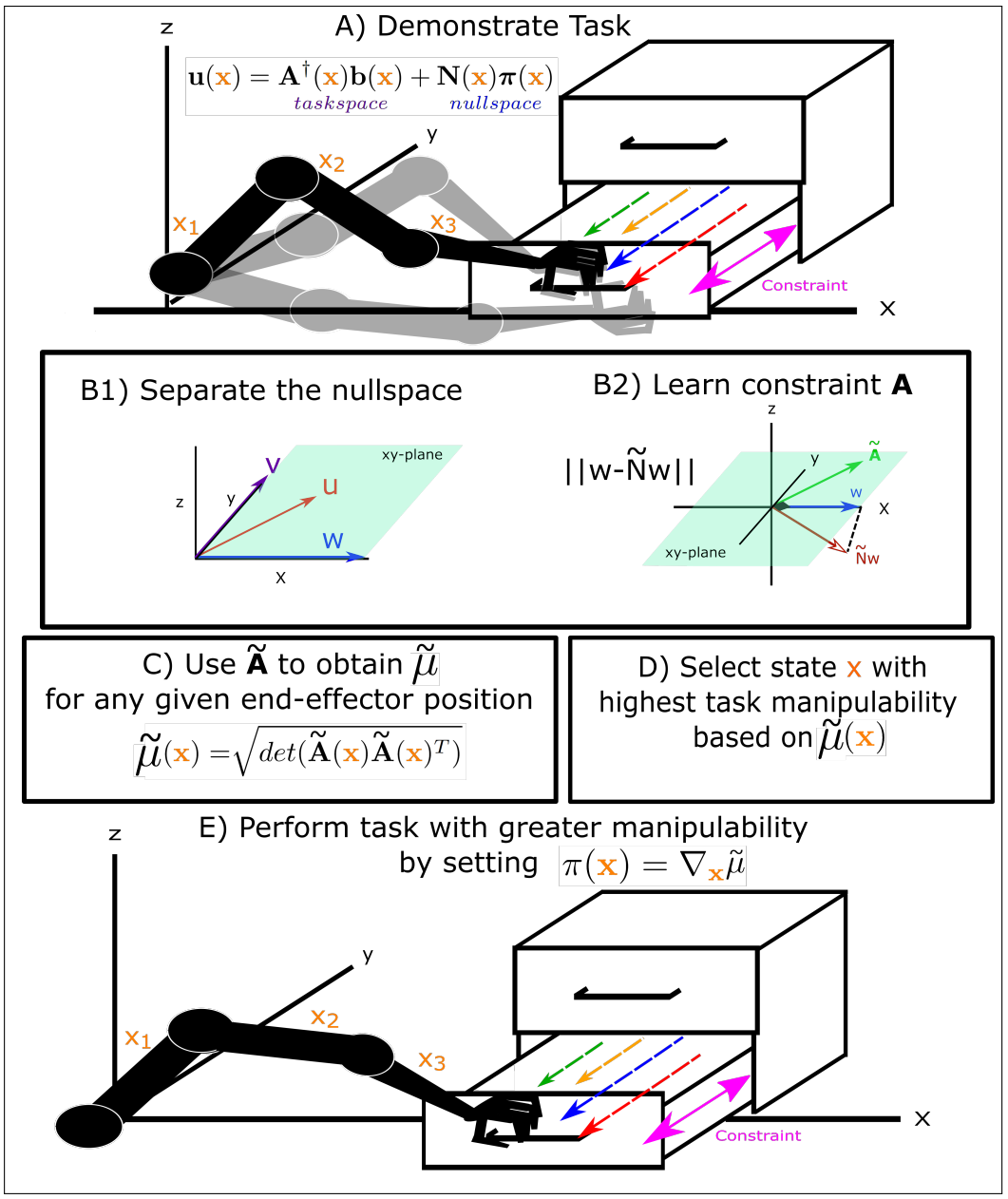


FIGURE 4.4: Overview of approach to maximising manipulability in programming by demonstration tasks. (A) Motion data is collected through demonstrations of the task. (B1) The data is used to determine the separate task and null space components so that (B2) the latter can be used to estimate the constraint. (C) Using the estimated constraint matrix, an estimate of the constrained manipulability $\tilde{\mu}$ is made. (D) This estimate is used to select states with greater manipulability, and (E) control the robot toward these when performing the primary task.

4.6 Experiments on Singularity Avoidance in the Joint-space

In this section, the proposed approach is first examined through a simulated 3-link planar system, before evaluating its performance in the context of programming by demonstration of a physical robot.[4]

4.6.1 Evaluation Criteria

Learning the constrained manipulability may not always be a trivial matter depending on the task at hand. Factors such as high dimensionality of a system or the structure of particular constraints in comparison to others, can lead to poor learning performance. It is therefore necessary to define a metric to assess learning performance.

The Normalised Manipulability Index Error (NMIE) evaluates the fitness of the learnt manipulability index $\tilde{\mu}$, and measures the distance between the true and learnt manipulability index

$$E_{NMIE} = \frac{1}{\mathcal{N}\sigma_{\mu}^{2}} \sum_{n=1}^{\mathcal{N}} ||\mu_n - \tilde{\mu}_n||^2 \tag{4.7}$$

where $\mathcal{N}$ is the number of data points. The error is normalised by the variance of μ. The MIE will reach zero as $\tilde{\mu} \to \mu$.

[4]The data supporting this research are openly available from King's College London at `DOI: 10.18742/RDM01-478`. Further information about the data and conditions of access can be found by emailing `research.data@kcl.ac.uk`

4.6.2 Simulated Three Link Planar Arm

The aim of the first evaluation is to test the robustness of learning the manipulability from motion data. The setup is as follows.

Constrained motion data is gathered from a kinematic simulation of a 3-link planar robot. The state and action space refer to the joint angle position and velocities, respectively, *i.e.,* $\mathbf{x} := \mathbf{q} \in \mathbb{R}^3$ and $\mathbf{u} := \dot{\mathbf{q}} \in \mathbb{R}^3$. The task space is described by the end-effector coordinates $\mathbf{r} = (r_x, r_y, r_\theta)^\top$ referring to the positions and orientation, respectively. The simulation runs at a rate of $50\,Hz$. Joint space motion of the system is recorded as it performs tasks under different constraints in the end-effector space. As described in Section 4.4, a task constraint at state $\mathbf{x}$ is described through

$$\mathbf{A}(\mathbf{x}) = \mathbf{\Lambda}\mathbf{J}(\mathbf{x}) \tag{4.8}$$

where $\mathbf{J} \in \mathbb{R}^{3\times3}$ is the manipulator Jacobian, and $\mathbf{\Lambda} \in \mathbb{R}^{3\times3}$ is the selection matrix specifying the coordinates to be constrained. The following three constraints are evaluated:

$$\begin{aligned}
&1) \quad \mathbf{\Lambda}_{x,y} = ((1,0,0),(0,1,0),(0,0,0))^T.\\
&2) \quad \mathbf{\Lambda}_{x,\theta} = ((1,0,0),(0,0,0),(0,0,1))^T.\\
&3) \quad \mathbf{\Lambda}_{y,\theta} = ((0,0,0),(0,1,0),(0,0,1))^T.
\end{aligned}$$

To simulate demonstrations of reaching behaviour, the robot end-effector starts from a point chosen uniform-randomly $q_1 \sim U[0°, 10°], q_2 \sim U[90°, 100°], q_3 \sim U[0°, 10°]$ to a task space target $\mathbf{r}^*$ and following a linear point attractor policy

$$\mathbf{b}(\mathbf{x}) = \mathbf{r}^* - \mathbf{r} \tag{4.9}$$

where $\mathbf{r}^*$ is drawn uniformly from $r_x^* \sim U[-1, 1], r_y^* \sim U[0, 2], r_\theta^* \sim U[0, \pi]$. Targets without a valid inverse kinematic solution are removed. All trajectories also use a

TABLE 4.1: NMIE (mean±s.d.)$\times 10^{-3}$ for each constraint over 50 trials.

Constraint	NMIE
$\mathbf{\Lambda}_{x,y}$	$0.001 \times 10^{-3} \pm 0.007 \times 10^{-3}$
$\mathbf{\Lambda}_{x,\theta}$	$0.007 \times 10^{-3} \pm 0.004 \times 10^{-2}$
$\mathbf{\Lambda}_{y,\theta}$	$0.002 \times 10^{-5} \pm 0.003 \times 10^{-5}$

point attractor as a control policy in $\mathbf{w}$

$$\boldsymbol{\pi}(\mathbf{x}) = \boldsymbol{\xi}^* - \mathbf{x}. \tag{4.10}$$

The latter enforces consistency, that makes it easier to separate the constraint from the control policy. $\boldsymbol{\xi}^*$ is arbitrarily chosen as $q_1 = 10°$, $q_2 = -10°$, $q_3 = 10°$. For each constraint, 100 trajectories are generated, with each trajectory containing 10 data points (1,000 points per constraint). This is repeated to get separate training and testing data sets (a total of 2,000 points per constraint). Finally, this whole experiment is repeated 50 times.

The NMIE is presented in Table 4.1. As can be seen, learning of the constrained manipulability index is successful with errors less than 10^{-5}. This shows that the manipulability index can be learnt with very high precision through demonstrations, without having to explicitly know how the constraints affect the system's motions.

To further assess the suitability of using $\tilde{\mu}$ instead of μ, the RMSE is evaluated for 20 randomly generated trajectories of 100 points using $\boldsymbol{\pi}_\mu$ learnt under the constraint $\mathbf{\Lambda}_{x,y}$ and $\boldsymbol{\pi}_{\tilde{\mu}}$ following Section 4.5. The starting point is chosen uniform-randomly $q_1 \sim U[0°, 10°], q_2 \sim U[90°, 100°], q_3 \sim U[0°, 10°]$, and following Equation 4.9, r^* is drawn uniformly from $r_x^* \sim U[-1, 1], r_y^* \sim U[0, 2], r_\theta^* \sim U[0, \pi]$. This produces two trajectories, one using $\boldsymbol{\pi}_\mu$ and the other $\boldsymbol{\pi}_{\tilde{\mu}}$. Results given as (mean±s.d.)$\times 10^{-4}$ are 2.069 ± 1.001. These errors being lower than 10^{-3} in both the mean and standard deviation indicate that all 20 trajectories are accurately reproduced, therefore $\boldsymbol{\pi}_{\tilde{\mu}}$ is an appropriate replacement for when $\boldsymbol{\pi}_\mu$ is difficult to infer.

At this point, the suitability of $\boldsymbol{\pi}_{\tilde{\mu}}$ has been established. In order to understand the benefit of using the manipulability-based controller (Equation 4.6) to handle redundancy in the joint space, its performance can be compared to that of other commonly used policies when encountering singularities in $\boldsymbol{\Lambda}_{x,y}$. As examples of the latter, a zero policy and a linear point attractor in $\boldsymbol{\pi}$ are chosen. The zero policy emulates the most common and simplest approach (being the shortest path in the joint space directly towards the target subject to the task constraints). The linear point attractor is also a common choice as a null space policy, as a way of bringing the arm to a default posture. When considering how systems behave near singular configurations, it is also important to consider the case where a system has an *ad hoc* way to deal with singularities, which is becoming more common among safety protocols in commercialised systems for novice users (see Section 4.2.3). Thus, two cases are presented here, one where a system starts in a singular configuration without any impromptu way of dealing with singular values, and one case where singularities are dealt with by setting the singular value to zero after it crosses a certain threshold. Here, Matlab's *ad hoc* approach is used, whereby the pseudoinverse function determines when singularities are encountered by using a threshold of `max(size(A))` $\times$ `eps(norm(A))`, which in this case is 1.332×10^{-15}. The `eps` in Matlab calculates the floating-point relative accuracy.

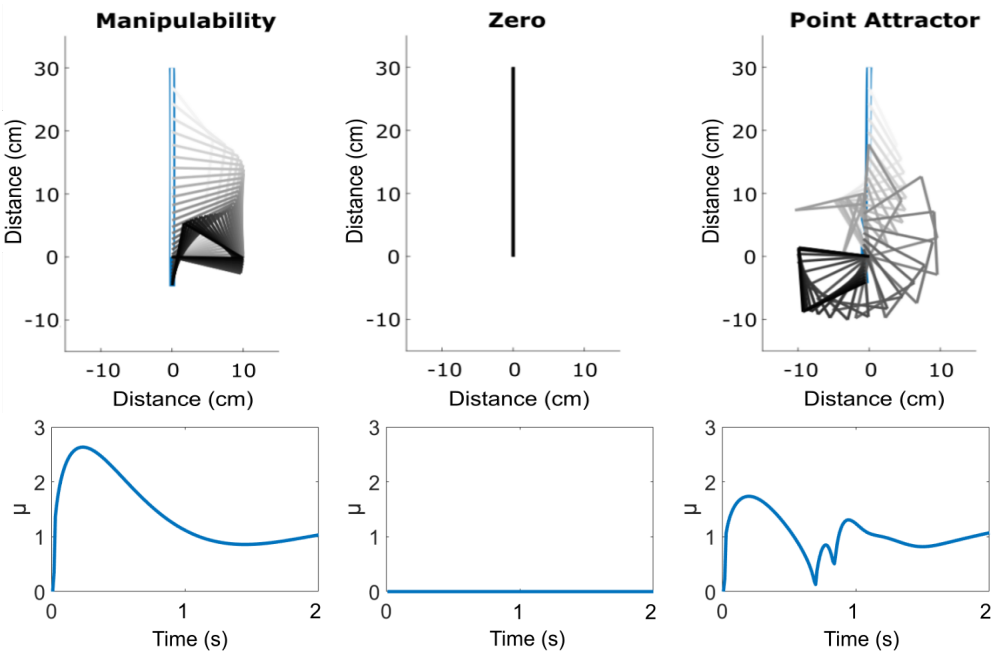


FIGURE 4.5: Comparing the manipulability, zero and point attractor in π where singular values in the task space are replaced with 0. The bottom row shows the manipulability over time of the corresponding systems in the top row throughout the trajectory.

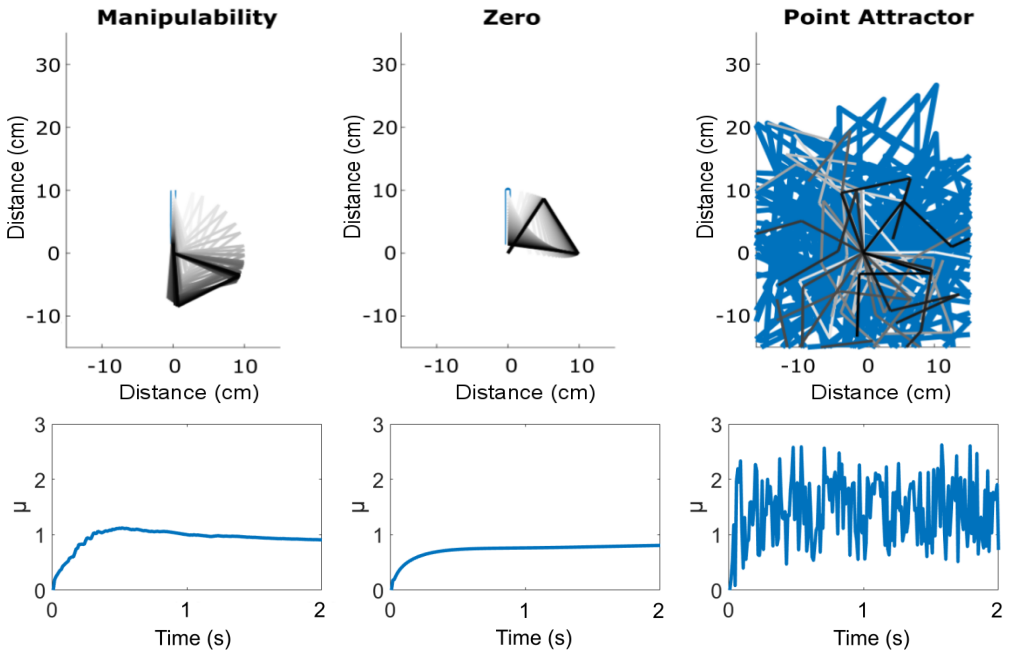


FIGURE 4.6: Comparing the manipulability, zero and point attractor in π where singularities are not dealt with. The bottom row shows the manipulability over time of the corresponding systems in the top row throughout the trajectory.

Figure 4.5 above shows how a system behaves under the three different control policies in $\boldsymbol{\pi}$. This system is subject to a task constraint in the r_x and r_y coordinate and uses three control policies to evaluate how each handles movement from a singular pose. As shown, the proposed method is able to move away from the singular starting pose $q_1 = (90+10^{-12})^\circ$, $q_2 = 360^\circ$and $q_3 = -360^\circ$ using the learnt manipulability. On the other hand, the zero policy gets stuck at the starting point as it attempts to move directly down towards the target $\mathbf{r}^* = (0, 0)$. The point attractor does succeed in the task, however, the movement to the default posture $q^* = (-190^\circ,9^\circ,-307^\circ)^T$ brings the robot close to singular configurations. While both the manipulability and point attractor policies reach their target, it is evident when looking at the bottom row that the (true) manipulability index of both systems are vastly different. As shown, the point attractor nearly approaches a singular configuration at around 0.7 seconds into the movement which explains the overall erratic movement to reach the target. On the other hand, the manipulability encounters no such problem as it moves towards the target while maximising its manipulability throughout the movement.

Figure 4.6 above looks at a case where no *ad hoc* method is used to detect singularities. In this case, a starting pose is set near[5] a singular pose of $q_1 = 90^\circ$, $q_2 = -180^\circ$and $q_3 = (-180 + 10^{-10})^\circ$. As shown, both the manipulaiblity and zero policy move away from the near-singular configuration. On the other hand, the point attractor with a default posture of $q^* = (-33^\circ,\text{-}283^\circ,193^\circ)^T$ exhibits highly unstable behaviour (its next state exceeds 10^9 for each joint). This type of unpredictable behaviour is the most hazardous when having systems work in the real world.

Overall, these experiments show that the constrained manipulability of a system can be learnt through programming by demonstration. Moreover, it is also evident that the learnt manipulability index is the preferable controller to avoid singularities in comparison to a straight-forward zero policy or a simple point attractor.

[5]Note that, starting in a singular configuration would lead to a division over 0 in the task space within the first step of the system's movement regardless of the control policy.

4.6.3 Real world 7-Link Sawyer Arm

The final experiment assesses the proposed approach in a real world task executed on the Sawyer, a 7DoF revolute physical robotic system with a maximum reach of 1260mm and precision of ±0.1mm. The experimental scenario chosen is the closing of a drawer.

The state and action space refer to the joint angle position and velocities, respectively, *i.e.,* $\mathbf{x} := \mathbf{q} \in \mathbb{R}^7$ and $\mathbf{u} := \dot{\mathbf{q}} \in \mathbb{R}^7$. The task space is described by the end-effector coordinates $\mathbf{r} = (r_x, r_y, r_z)^\top$ referring to positions in 3D cartesian space. The system runs at a rate of $100\,Hz$. Joint space motion is kinaesthetically recorded by guiding the Sawyer as it performs tasks under a constraint in the end-effector space. $\mathbf{J} \in \mathbb{R}^{3\times7}$ is the manipulator Jacobian and is assumed known *a priori*. $\mathbf{\Lambda} \in \mathbb{R}^{3\times3}$ is the selection matrix specifying the coordinates to be constrained. The following constraint is evaluated which models a drawer when it is orientated such that the constraint's null space lies along the x-axis (as shown in Figure 4.7 below)

$$\mathbf{\Lambda}_x = ((1,0,0),(0,0,0),(0,0,0))^T.$$

In order to have consistency in $\boldsymbol{\pi}$, the system starts in a default pose of $q_1 \sim -100°$, $q_2 \sim 30°$, $q_3 \sim -100°$, $q_4 \sim 40°$, $q_5 \sim -60°$,$q_6 - 70°$, $q_7 \sim 250°$ where the joints point outward, such that stretching out the system's arm away from its body and along the constraint resolves redundancy in a similar manner for each trajectory.

Fifty trajectories are recorded by a user familiar with the system through kinaesthetic demonstrations. In each trajectory, movement consists of the aforementioned default start pose to a constrained pose where the drawer is being closed (containing both task and null space). The data is down-sampled by a rate of 200 such that each trajectory is reduced to 6-14 points. This is done as the direction of the constrained movements are captured even with such little data, and more data simply results in longer computation times.

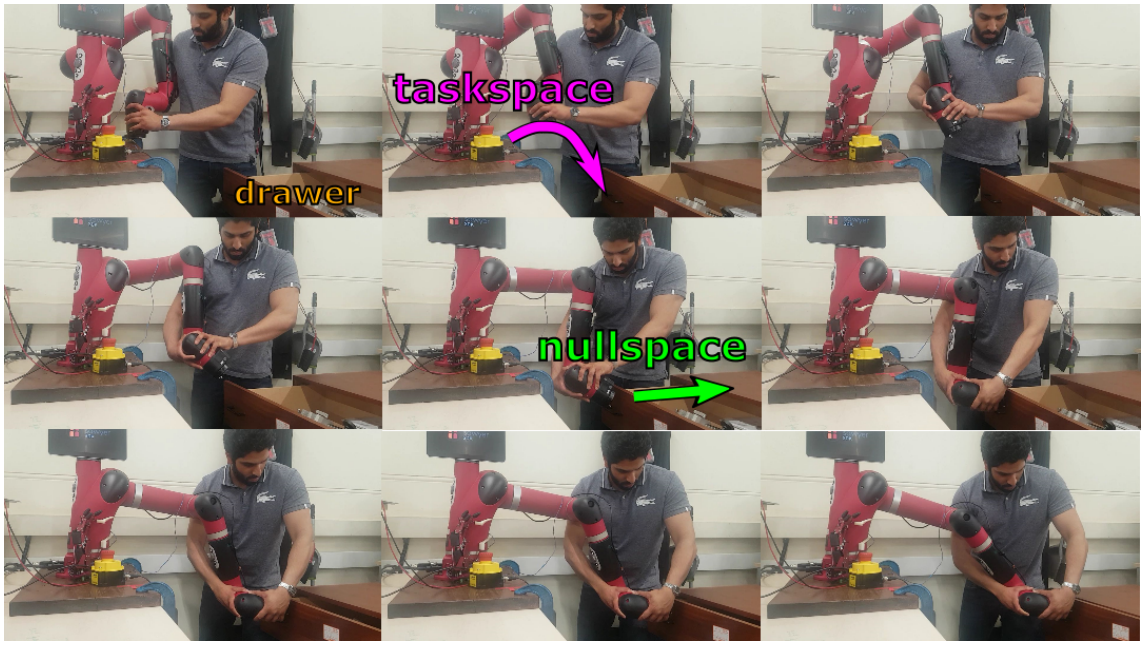


FIGURE 4.7: Learning the task constraint when closing the drawer through programming by demonstration using the Sawyer. The null space lies in the direction of the drawer being closed and the task space is perpendicular to this.

The MIE learnt from 50 trajectories is 0.002, therefore it is learnt successfully with errors[6] below 10^{-2}.[7] While it is shown that the manipulability index is learnt, it is important to establish whether the estimated manipulability is still suitable as a cost function to avoid singularity with its greater error margin in comparison to the simulated 3DoF system. To this end, the RMSE is evaluated for 20 randomly generated trajectories of 100 points using the learnt model. The starting point is chosen uniform-randomly $q_1 \sim U[0°, 10°], q_2 \sim U[90°, 100°], q_3 \sim U[0°, 10°], q_4 \sim U[90°, 100°], q_5 \sim U[0°, 10°], q_6 \sim U[90°, 100°], q_7 \sim U[0°, 10°]$. r^* is drawn uniformly from $\sim U[-1, 1]$ for the x-axis. Two trajectories are produced, one using $\boldsymbol{\pi}_\mu$ and the other $\boldsymbol{\pi}_{\tilde{\mu}}$. The results are 0.220 ± 0.094 (mean±s.d.). Considering the high dimensionality of the robot, it is reasonable to assume an increased error in comparison to the simulated 3-link system.

[6] In a similar experiment where the null space of the constraint lies along the y-axis, the performance was alike.

[7] The variables of the learning algorithm can be adjusted trading speed for accuracy depending on the intended use (see Section 3.4).

4.7 Extended Analysis of Singularity Maps and Avoidance

The previous set of experiments focus on joint-space singularities which can in most cases be overcome though manipulability based redundancy resolution. However, constraints can also happen in the end-effector space if the system is driven through certain points regardless of its joint-space configuration. This is covered in the following sections.

4.7.1 Example: Hose Grasping

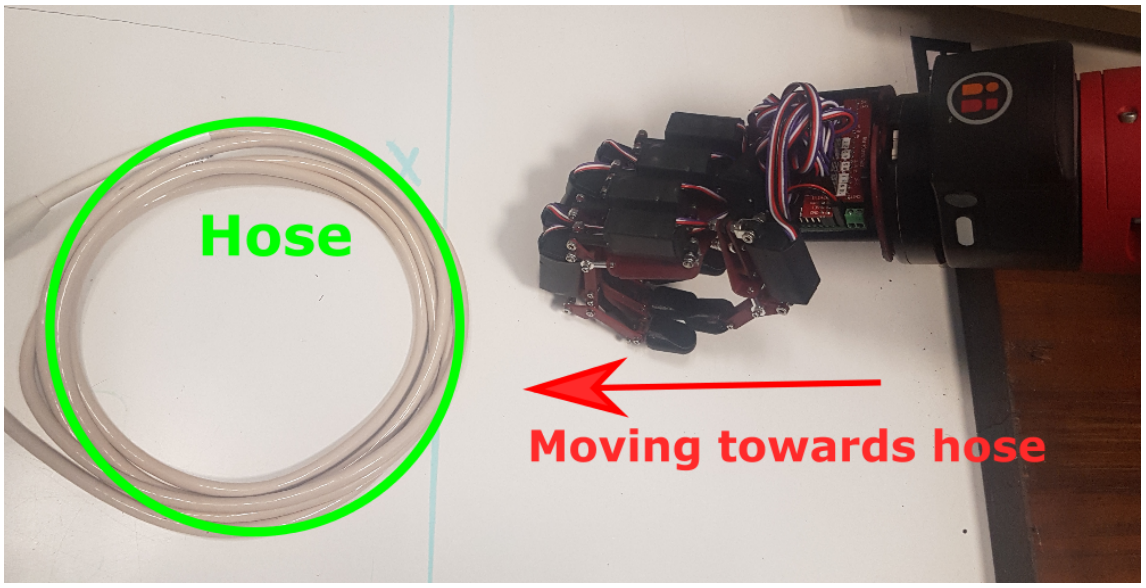


FIGURE 4.8: Moving to grasp a hose which has a state-dependent circular polynomial constraint.

As a real world problem, consider placing the robot within a dynamic environment, such as firefighters responding swiftly to an emergency. A robot trained to work in this environment, in this case a storeroom, must gather equipment, such as a fire hose, without delay and interference to the firefighters. In this case, consider the

problem of teaching the robot to grasp a hose through programming by demonstration (as shown in Figure 4.8 above). Programming by demonstration is suitable because the robot is in a familiar environment and once trained it can assemble and disassemble the fire equipment while the firefighters focus on the emergency. Then, the primary task policy is to bring the robot end-effector to a point on the hose, where the constraint matrix $\mathbf{A}$ is determined by the shape of the hose. If the hose is stored ineffectively or needs to be collected from the fire engine, it may have a complex state-dependent polynomial form (Figure 4.9-A), or it can also be wrapped up neatly in loops such that the polynomial constraint forms a circular shape. Alternatively, the hose can be partly hanging off a wall, or stretched out during use giving it a state independent linear shape (Figure 4.9-B).

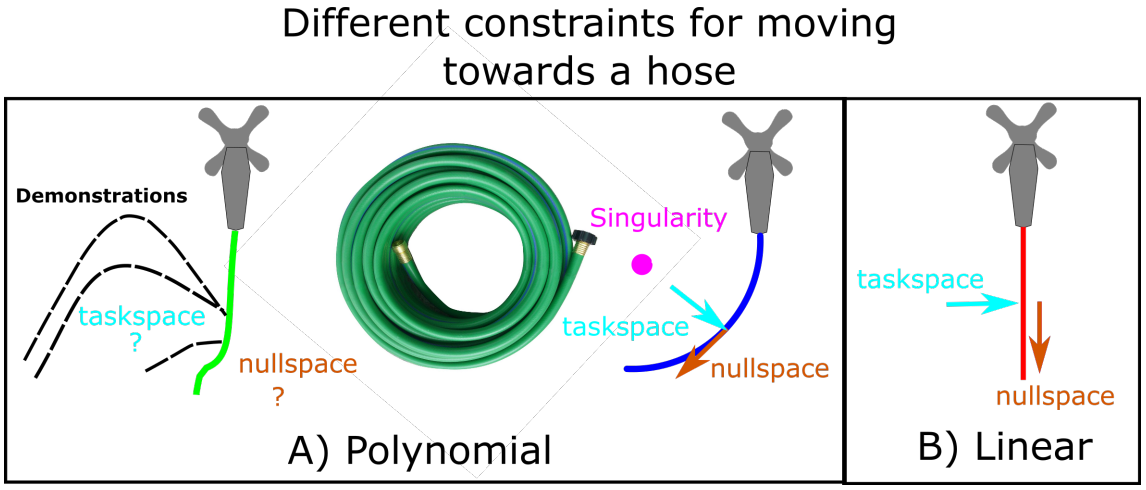


FIGURE 4.9: Demonstrating to learn different constraints. The demonstrations have a task space of moving to the target constraint at different speeds and a null space defining how to approach the target

Where the task is kinesthetically demonstrated, the user might manually guide a robot from a random point in the work space to the hose, and repeat this action multiple times using different starting points and varying speeds. In this scenario, the primary task might be learnt in a straightforward manner using one of several constraint learning methods (see Section 4.4).

Moreover, unless explicitly instructed, the user might not take specific care of how the motion appears in the null space (*i.e.,* the space orthogonal to the primary task dimension) when performing those demonstrations. For example, the user might choose to move along the shortest straight line distance to the hose, or through a natural arc in line with the joint structure of either their arm or that of the robot. It is unlikely that an average user will know to avoid unstable or singular configurations—however, these may occur in unexpected locations. For example, if the hose is wrapped in a circle (Figure 4.9-A above), a singularity appears at the centre of the circle, potentially creating a hazard during playback of the motion. More generally, the hose might be hanging in an uncertain shape (see Figure 4.9-A above), making the constraint and therefore the landscape of possible unstable points difficult to infer in analytical form.

Figure 4.10 below shows a brief overview of the major steps involved in the proposed approach. The approach follows the same structure as discussed in Section 4.2.3 to Section 4.5.

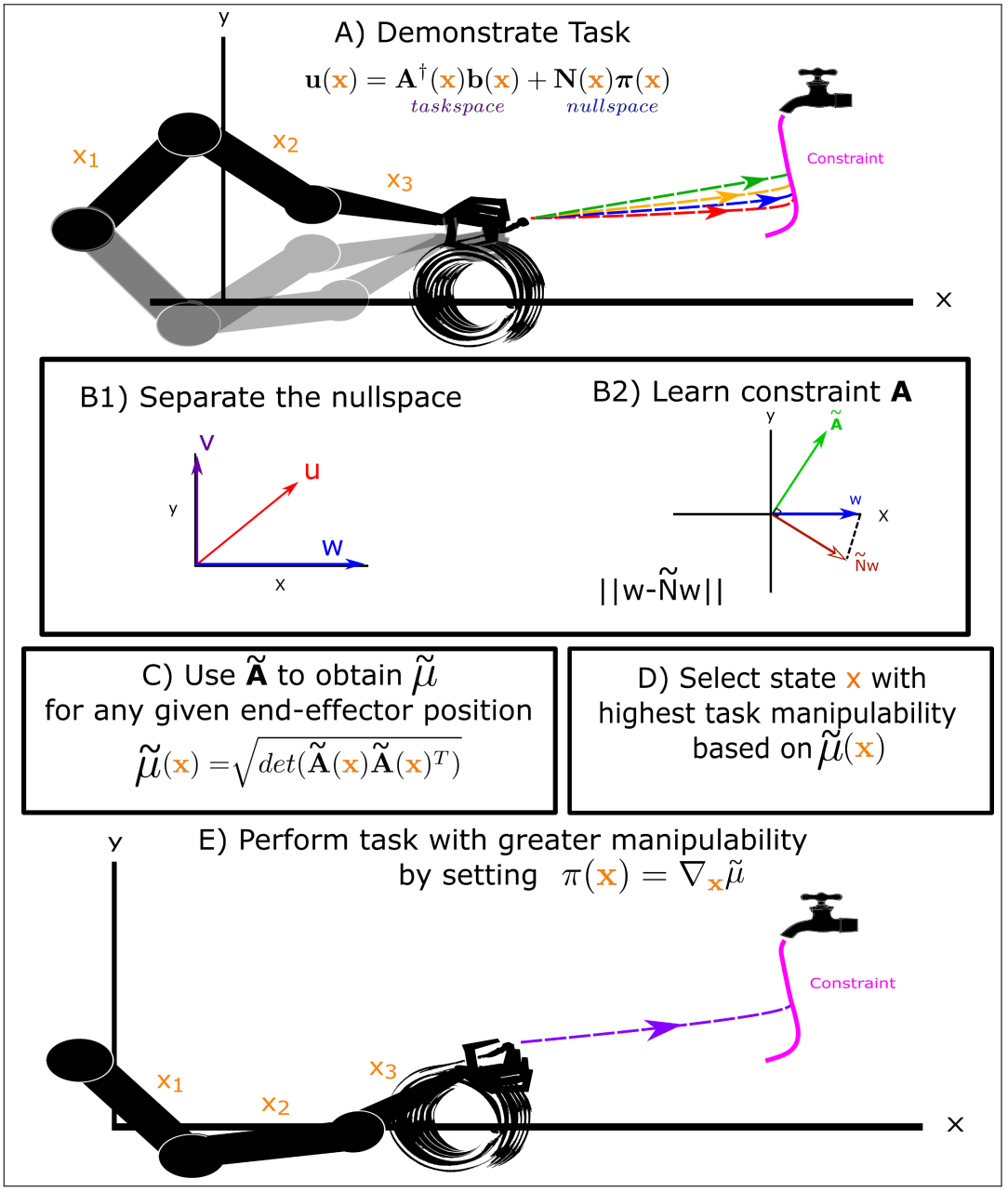


FIGURE 4.10: Overview of approach to maximising manipulability in programming by demonstration tasks. (A) Motion data is collected through demonstrations of the task. (B1) The data is used to determine the separate task and null space components so that (B2) the latter can be used to estimate the constraint. (C) Using the estimated constraint matrix, an estimate of the constrained manipulability $\tilde{\mu}$ is made. (D) This estimate is used to select states with greater manipulability, and (E) control the robot toward these when performing the primary task.

4.8 Experiments on Singularity Avoidance in the End-effector Space

In this section, the proposed approach is first examined through a simple 2D simulation, before evaluating its performance in the context of programming by demonstration of a physical robotic system.

4.8.1 2D Simulation

The aim of the first evaluation is to demonstrate the use of a learnt manipulability map in the context of teaching a robot by demonstration to grasp a hose while avoiding singular points. The set up is as follows.

Constrained motion data is gathered from a kinematic simulation of a 2-link planar robot. The state and action space refer to the end-effector position and velocities, respectively, *i.e.,* $\mathbf{x} = (r_1, r_2)^\top$ and $\mathbf{u} = (\dot{r}_1, \dot{r}_2)^\top$. The simulation runs at a rate of $50\,Hz$.

The robot's motion is subject to the task constraint that depends on the configuration of the hose. The simulated situation looks at the hose scattered across the floor (*e.g.,* if no regard is given to how it is stowed away), giving it a complex state-dependent polynomial form. The ground truth constraint is therefore

$$\mathbf{A} = (2x_1^3 + 0.9x_1^2 + 0.2, 0.2x_2 + 0.7) \tag{4.11}$$

(the coefficients of this polynomial are arbitrarily chosen).

For learning, we use Equation 4.3, which makes $\mathbf{\Phi}$ known *a priori*. The first row of $\mathbf{\Phi}$ is the ground truth and the second row of $\mathbf{\Phi}$ is a linear constraint $(1, 0)$, simulating the hose hanging off a wall.

To simulate demonstrations of reaching behaviour, a point attractor policy

$$\mathbf{b}(\mathbf{x}) = \rho(\tau^* - \tau) \tag{4.12}$$

is used. It brings the robot end-effector from a starting point chosen uniform-randomly ($(\mathbf{x}_0)_i \sim \mathcal{U}(0,1), i \in \{1,2\}$) to a point on the hose. Here, ρ controls the reaching speed and τ^* is the target point located anywhere on the hose. To simulate variations in speed across the 1000 demonstrations used in this evaluation, ρ is drawn uniform-randomly *i.e.,* $\rho \sim \mathcal{U}(0,1)$. τ^* is arbitrarily chosen to be 5.

To emulate how different people may perform the same task in different ways, the set of 50 trajectories are repeated 20 times using a point attractor as a control policy in $\mathbf{w}$

$$\boldsymbol{\pi}(\mathbf{x}) = 0.1(\xi^* - \mathbf{x}) \tag{4.13}$$

ξ^* is selected uniform-randomly across the space ($(\xi^*)_i \sim \mathcal{U}(0,2), i \in \{1,2\}$), this gives a shape to the demonstrator's motion. It emulates variances between people, through a secondary intermediate target unrelated to the hose. Only the first 10 points of each trajectory are used for learning, as the required information being the direction is captured. More data simply results in longer computation times. This produces 1000 trajectories (a total of 10,000 points) which is split equally into training/test data. Finally, this whole experiment is repeated 20 times for evaluation of the normalised projected policy error (NPPE) and normalised projected observation error (NPOE) [13] (defined in Appendix A). Results in predicting the polynomial constraint are (mean±s.d) over 20 trials with a NPPE of $0.016 \pm (6.632 \times 10^{-4})$ and NPOE of $0.005 \pm (1.980 \times 10^{-4})$. This shows successful learning of the constraint.

Once the selection matrix which picks the polynomial is learnt constraint, the constraint is used in the null space component as a controller which resolves redundancy by estimating manipulability, such that it moves away from singular points. This

is done following Section 4.5. The task space component simply uses the original policy extracted from the demonstrations according to Section 4.3.2.

To assess the suitability of using $\tilde{\mu}$ instead of μ for the case where the latter is difficult to infer, the first experiment compares how $\boldsymbol{\pi}_{\tilde{\mu}}$ and $\boldsymbol{\pi}_{\mu}$ perform.

To evaluate the performance of the paths taken, the following is proposed:

$$\gamma = \frac{1}{N} \sum_{n=1}^{\mathcal{N}} \mu \tag{4.14}$$

This normalised task manipulability measure (γ) evaluates the average task manipulability (Equation 4.5) over the entire trajectory.

The mean of Equation 4.14 is used to evaluate $\boldsymbol{\pi}_{\tilde{\mu}}$ and $\boldsymbol{\pi}_{\mu}$ over 20 trials. Resulting from the experiment using the polynomial constraint as the ground truth, $\boldsymbol{\pi}_{\tilde{\mu}}$ has an average score of 51.416 whereas $\boldsymbol{\pi}_{\mu}$ is 50.023. This shows that $\boldsymbol{\pi}_{\tilde{\mu}}$ and $\boldsymbol{\pi}_{\mu}$ result in similar scores indicating that $\boldsymbol{\pi}_{\tilde{\mu}}$ can be used instead of $\boldsymbol{\pi}_{\mu}$. To evaluate this further, Figure 4.11 below shows the qualitative difference between paths generated through $\boldsymbol{\pi}_{\tilde{\mu}}$ and $\boldsymbol{\pi}_{\mu}$. Visually there is a slight difference between both control policies, however they result in the same general shape and direction which indicates that $\boldsymbol{\pi}_{\tilde{\mu}}$ is an appropriate replacement for when $\boldsymbol{\pi}_{\mu}$ is difficult to infer.

At this point, the suitability of $\boldsymbol{\pi}_{\tilde{\mu}}$ over $\boldsymbol{\pi}_{\mu}$ has been established. Now, the following experiment evaluates the efficiency of the manipulability based controller (Equation 4.6) in comparison with using a linear point attractor in $\boldsymbol{\pi}$ as well as a zero policy. The linear point attractor following Equation 4.13 emulates a natural curve occurring when a person demonstrates an indirect reaching task, the zero policy on the other hand emulates the most common approach being the shortest path directly towards the task.

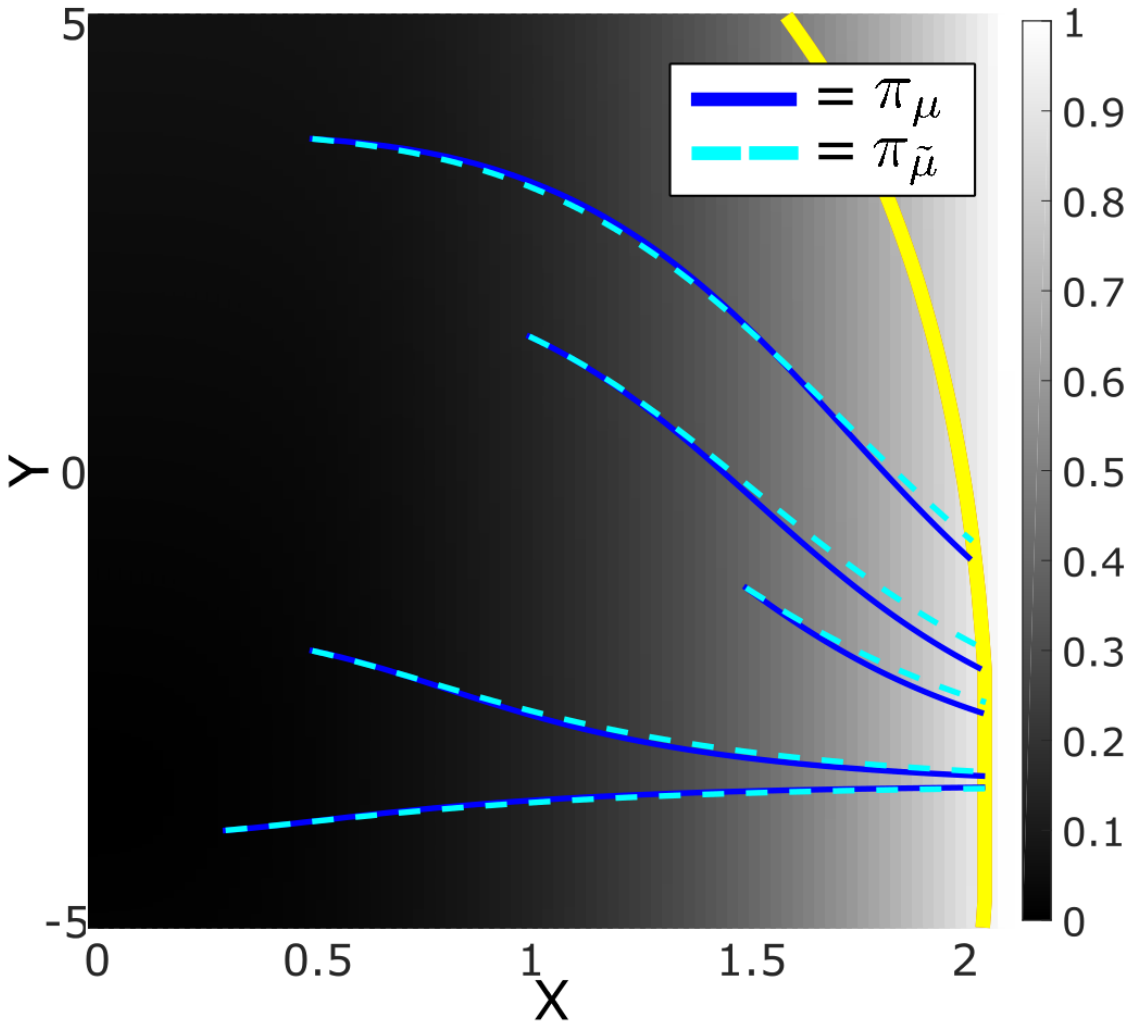


FIGURE 4.11: Comparing $\boldsymbol{\pi}_\mu$ and $\boldsymbol{\pi}_{\tilde{\mu}}$ in 5 randomly generated trajectories for each reaching the hose (yellow line). The manipulability index of the system under the polynomial constraint is scaled to range from 0 (singularity) to 1 (being the highest manipulability within support of the data), this is visually presented as a gradient between black and white, respectively.

To compare these control policies, Equation 4.14 is used to measure the system's manipulability throughout the movement. Moreover, the experiment is conducted over 20 trials for each stated control policy.

One of the paths taken for each different $\boldsymbol{\pi}$ is shown in Figure 4.12 below. Starting from the same point, the zero policy moves directly towards the target in the task space, while the longest route is taken by the point attractor which noticeably makes

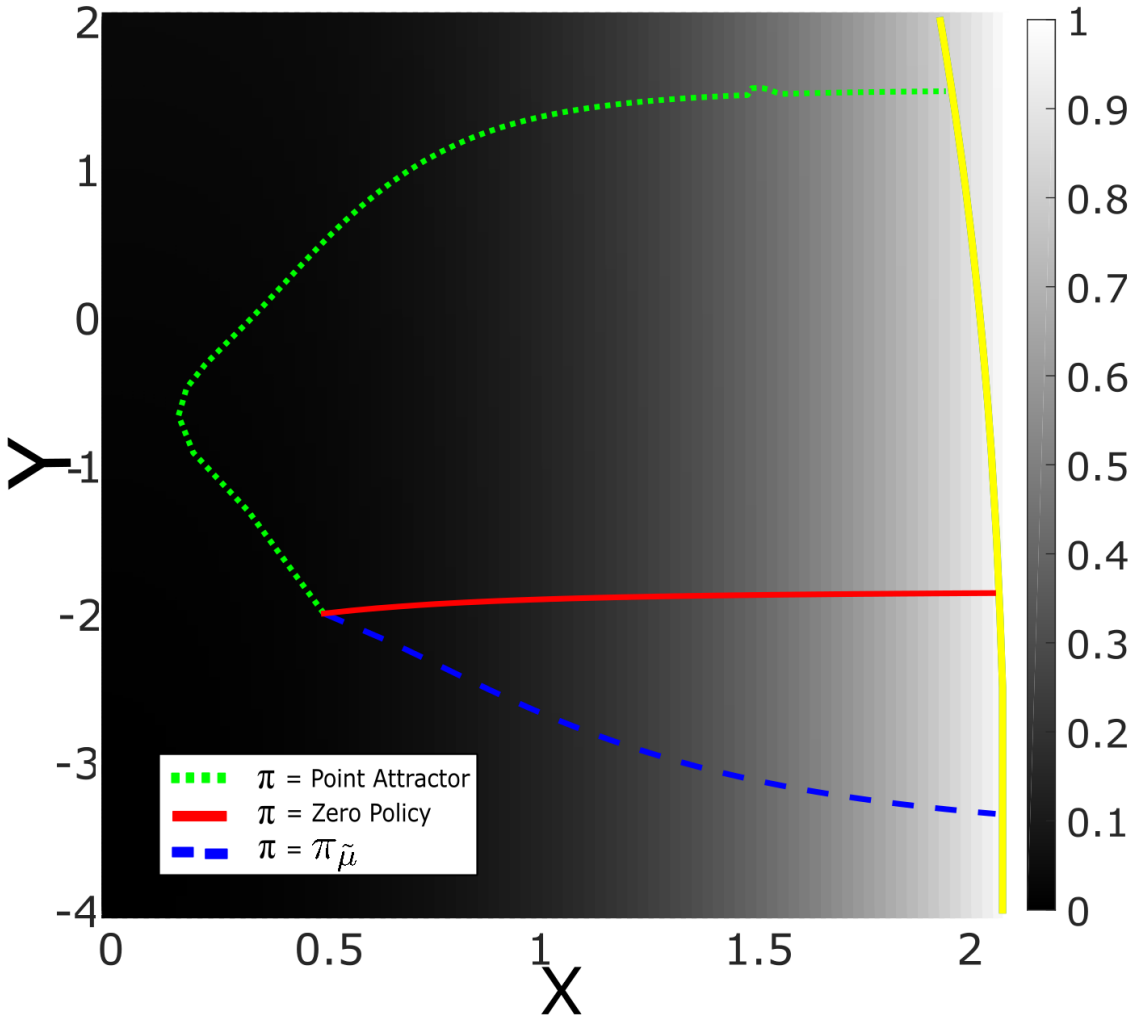


FIGURE 4.12: Sample of comparing different control policies using the learnt constraint taken from one trial

unnecessary movements. In the real world, this excess movement would result in longer times to complete a task.

Results concluding from the experiment give manipulability average scores (Equation 4.14) of 61.415, 59.459 and 60.724 for $\tilde{\mu}$, the point attractor and zero policy, respectively. As expected, using the learnt selection matrix with the correct constraint as a cost function (Equation 4.6) results in the highest average manipulability throughout the trajectory. While it does not take the shortest path (as done with

the zero policy), using the cost function results in a greater distance from the singular region as indicated in the score using Equation 4.14. This difference in γ is important, as a greater distance from singular regions minimises the risk of crossing the singular point while the system autonomously performs a task, thereby reducing the chance of encountering unpredictable behaviour.

Figure 4.13 shows an example of how a system may behave when dangerously close to the singular point in its end-effector space. As shown by the green dashed line,

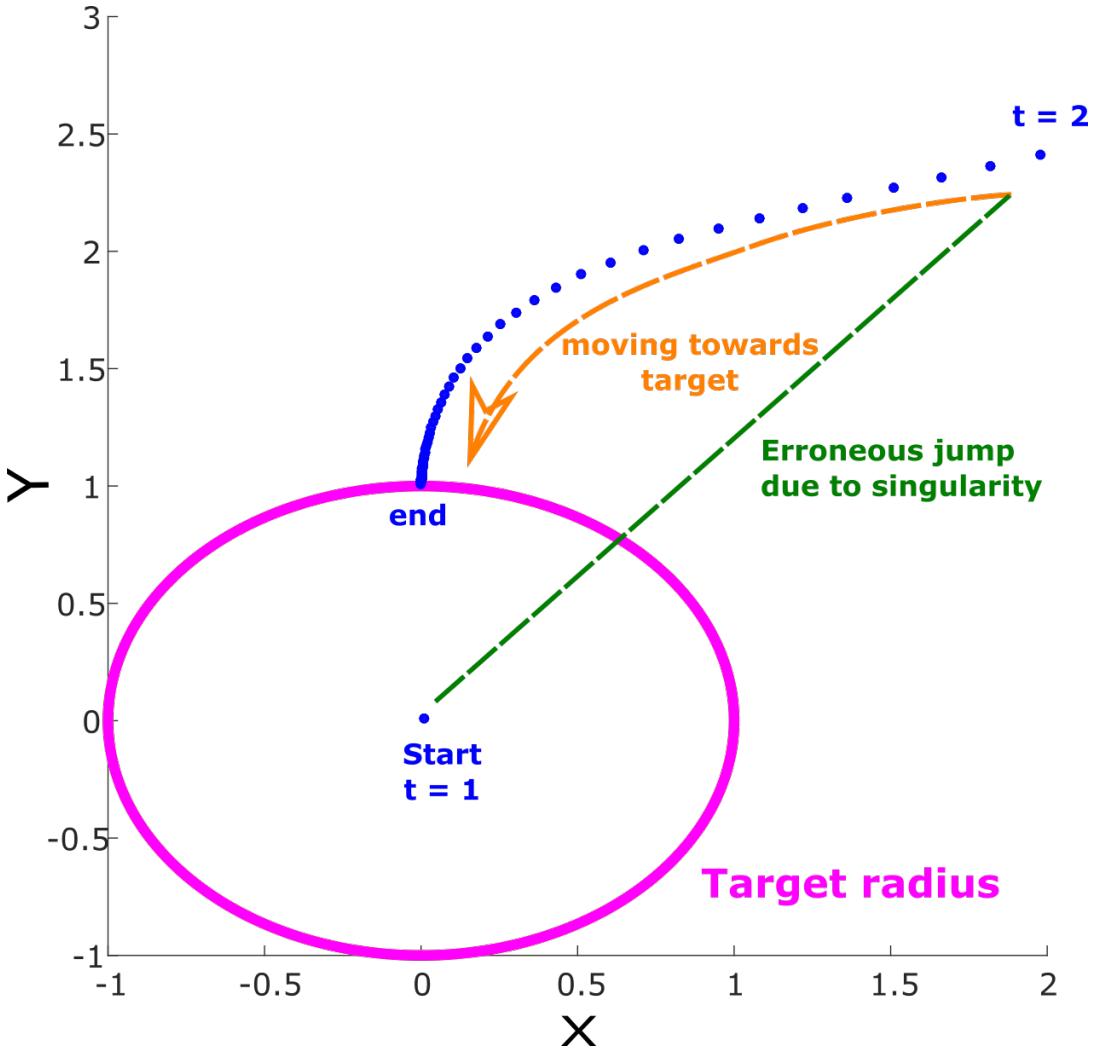


FIGURE 4.13: Sample erroneous behaviour when attempting moving from close to a singular point (centre of the circle) to any point on the target radius. After the first step in time (t),the target is surpassed by an unpredictable distance.

the primary task of reaching the target radius is completely overshot without taking any intermediate steps in-between, where the system jumps from close to the centre of the circle (in a singular region) to far beyond the indicated target line. This unpredictable behaviour not only affects the system from performing its task but also makes it difficult for the user to know where the system may attempt to move. In this case, it unnecessarily moves over twice the distance required to reach the target.

4.8.2 3D Simulation

This experiment aims to evaluate how well the learning method performs when demonstrating the task of grasping a hose in a 3-dimensional setting. It uses a more complex selection of rows for the constraints, emulating for example if the constraint models a hose where one end is attached to a hydrant or runs over/under obstacles.

In this experiment, constrained motion data from a kinematic simulation of a robot moving in a 3D space is used. In this case, the state and action space refer to the end-effector position and velocities, respectively, *i.e.,* $\mathbf{x} = \mathbf{r} = [x_1, x_2, x_3], \mathbf{u} = \dot{\mathbf{r}} \in \mathbb{R}^3$.

Constraints imposed on the system are based on $\mathbf{A}(\mathbf{x}) = \mathbf{\Lambda}\mathbf{\Phi}(\mathbf{x})$ where $\mathbf{\Lambda} \in \mathbb{R}^{1\times 3}$ and $\mathbf{\Phi}(\mathbf{x}) \in \mathbb{R}^{3\times 3}$ is defined by the following 3 rows of constraints:

1. Polynomial constraint: $((4ax_1^3{+}3bx_1^2{+}2cx_1{+}d), (4ex_2^3{+}3fx_2^2{+}2gx_2{+}h), (4ix_2^3{+}3jx_2^2 + 2kx_2 + l))$
2. Circular constraint: $((2x_1), (2x_2), (2x_3))$
3. Linear constraint: $((1), (0), (0))$

In addition to the constraints from the previous experiment, a circular constraint in also included to emulate the most common way the hose is packed away.

Following the same steps as the previous experiment, for each set of 50 trajectories, the null space component is learnt with the same target for the point attractor $\boldsymbol{\pi}$

Constraint	NPPE	NPOE
100	0.002 ± 0.000	0.001 ± 0.000
010	0.064 ± 0.009	0.016 ± 0.002
001	0.093 ± 0.011	0.035 ± 0.004

TABLE 4.2: Normalised PPE and POE in predicting the projection matrix in the end-effector space. Results are (mean± s.d) over 20 trials

set uniform-randomly across the space $(\mathbf{x}_{pTarget})_i \sim \mathcal{U}(0,1), i \in \{1,2,3\}$ and with the first 10 points per trajectory, where a fixed time step κ of 0.02 is used between observed actions. Similar to how people would demonstrate a task repeatedly, the speed ρ of reaching the target (given in $\mathbf{b}$ from the task space) is also uniformly random across the space $(\rho)_i \sim \mathcal{U}(0,1), i \in \{1\}$ in all trajectories. All other elements in $\mathbf{b}$ follow the same structure as the previous experiment, with the addition of a circular constraint which has a target radius τ^* of 5 units. This results in a total of 5,000 points for learning as well as the same procedure for testing. Finally, to evaluate the NPPE and NPOE , this whole experiment is repeated 20 times.

Table 4.2 shows the NPPE and NPOE for learning the three constraints. These results show successful learning of the aforementioned constraints. While the circular and linear constraints are learnt with a similar accuracy, the polynomial constraint was learnt with the highest accuracy. This can be due to the higher order polynomial taking on a more complex shape in the 3D environment making it the most distinct.

Figure 4.14 below shows a comparison between the true manipulability map and a sample map constructed from one of the learnt trials. As shown, although the learnt map overestimates the regions of low manipulability, both plots are very similar and correctly display which regions in general are of higher or lower manipulability, even under a marginal error of the learnt constraint. Therefore, if a system has to avoid singularities during a task, this approach is viable even if the constraint isn't perfectly captured, as the overall direction towards manipulability maximisation can be ascertained. As explained in Section 4.4, displaying the manipulability map for a linear constraint is redundant due to its state-independence.

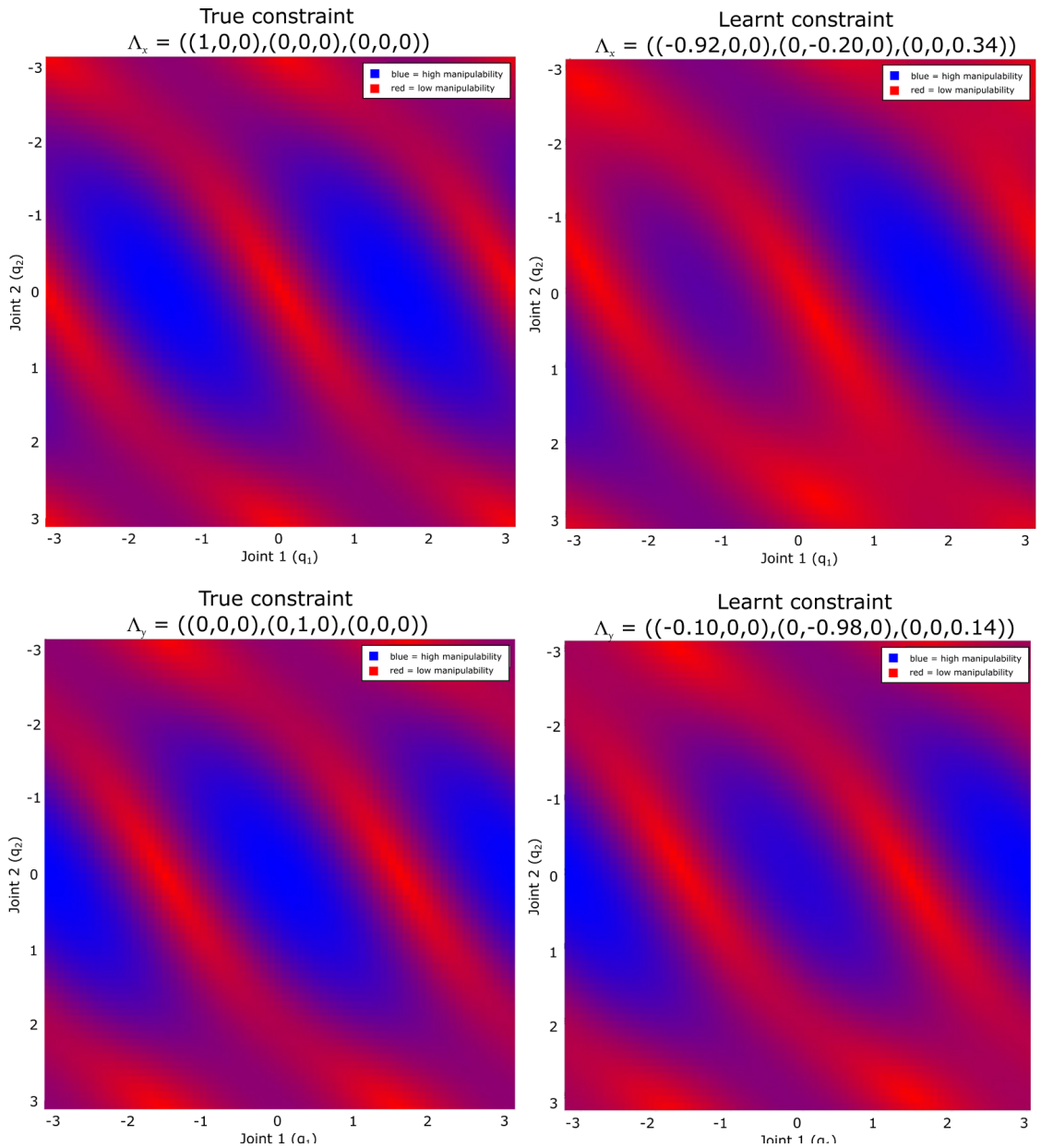


FIGURE 4.14: Comparing the manipulability Map for true and learnt constraints.

The experiment looks at using the polynomial constraint as the ground truth when comparing $\boldsymbol{\pi}_{\mu}$ and $\boldsymbol{\pi}_{\tilde{\mu}}$ in a 3D space. The mean of Equation 4.14 is used to evaluate $\boldsymbol{\pi}_{\tilde{\mu}}$ and $\boldsymbol{\pi}_{\mu}$ over 20 trials. $\boldsymbol{\pi}_{\mu}$ has an average score of 96.015 whereas $\boldsymbol{\pi}_{\tilde{\mu}}$ is 96.844. The similar average manipulability obtained over the course of each trajectory for $\boldsymbol{\pi}_{\mu}$ and$\boldsymbol{\pi}_{\tilde{\mu}}$ is in line with the previous 2D experiment, further demonstrating that it is an appropriate replacement for when $\boldsymbol{\pi}_{\mu}$ is not available.

$\boldsymbol{\pi}$	learnt γ
$\tilde{\mu}$	96.843
Point attractor	73.379
zero policy	90.735

TABLE 4.3: Comparing Task Manipulability as $\boldsymbol{\pi}$ vs other control policies.

The next experiment evaluates the performance of our task manipulability index in comparison to other policies, also using Equation 4.14 over 20 trials for each control policy. Similar to the 2D experiment, $\tilde{\mu}$ is compared as a manipulability function used in Equation 4.6 to a zero policy and a point attractor with the secondary task of moving to a uniform-randomly chosen point across the space where $(\mathbf{pTarget})_i \sim \mathcal{U}(0,2), i \in \{1,2,3\}$. The experiment is concluded with the results given in Table 4.3 above. As shown, using $\tilde{\mu}$ for optimising the trajectory based on manipulability clearly results in highest average manipulability throughout its motion. There is a greater difference in the γ scores between all three policies, with the point attractor achieving a significantly worse score. This can be attributed to the randomly selected secondary targets. Overall, these results are in line with the previous experiment, proving further that the use of $\tilde{\mu}$ at its worst performs as well as a zero policy, and in all other cases results in a greater locally optimised average manipulability index throughout its movement. Compared to other control policies, using $\tilde{\mu}$ minimises the risk of unpredictable behaviour from crossing the singular point while moving towards the target.

4.8.3 Real world 7-Link Sawyer Arm

The aim of this evaluation is to demonstrate the use of a manipulability map learnt through programming by demonstration, to avoid singularity in a real world environment.

Constrained motion data from the 7-link Sawyer robot is used. The state and action space refer to the end-effector position collected from the Sawyers sensors and velocities calculated from the position data, respectively, *i.e.,* $\mathbf{x} = \mathbf{r} = [r_1, r_2], \mathbf{u} = \dot{\mathbf{r}} \in \mathbb{R}^2$ at a sampling rate of $100\,Hz$.

Following the set up of the 2D simulation, the hose has a complex state-dependent polynomial form (Equation 4.11) chosen as the true constraint. For learning, we use the model defined in Equation 4.3 where the first row of $\mathbf{\Phi}$ is the ground truth and the second row of $\mathbf{\Phi}$ is a linear constraint $(1, 0)$. The route to reaching the constraint, to estimate $\mathbf{\Lambda}$ which selects the polynomial constraint, is at the discretion of the demonstrator. Figure 4.15 shows a few of many possible trajectories.

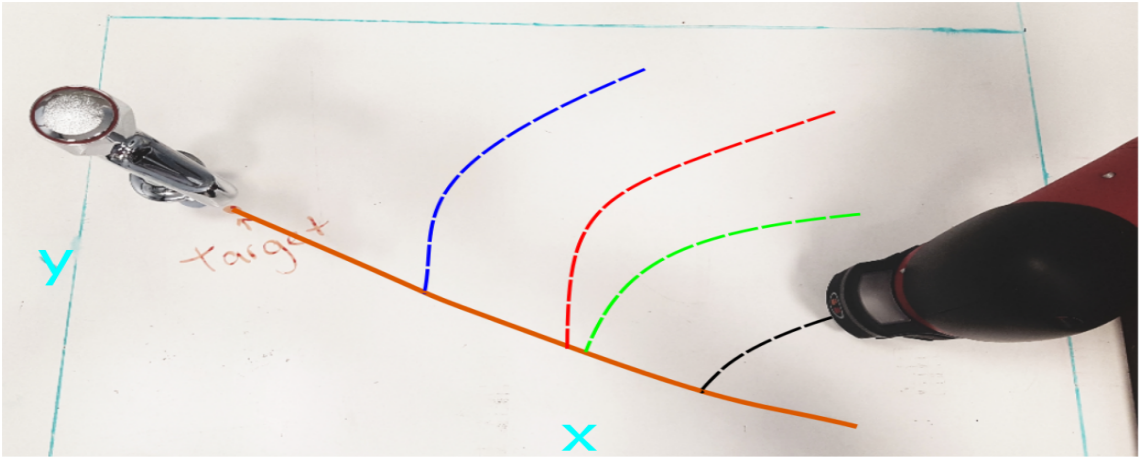


FIGURE 4.15: Experiment set up using the Sawyer with sample paths overlaid

Data sets consist of 3 sets of 10 trajectories, where each set uses a different control policy in the null space, however this is unknown in the real world with the human demonstrator. Again, to reduce unnecessary computation times, only a subsection of each trajectory is used for learning. All raw trajectories contain approximately

between 500 and 1000 points depending on their length as well as differences in the velocity due to a human element. After collection, the data is subsampled such that each trajectory consists of 10 equally spaced points. Figure 4.16-A presents a sample trajectory of the system reaching the polynomially constrained target hose, whereas Figure 4.16-B shows a self-collision caused when encountering a singularity.

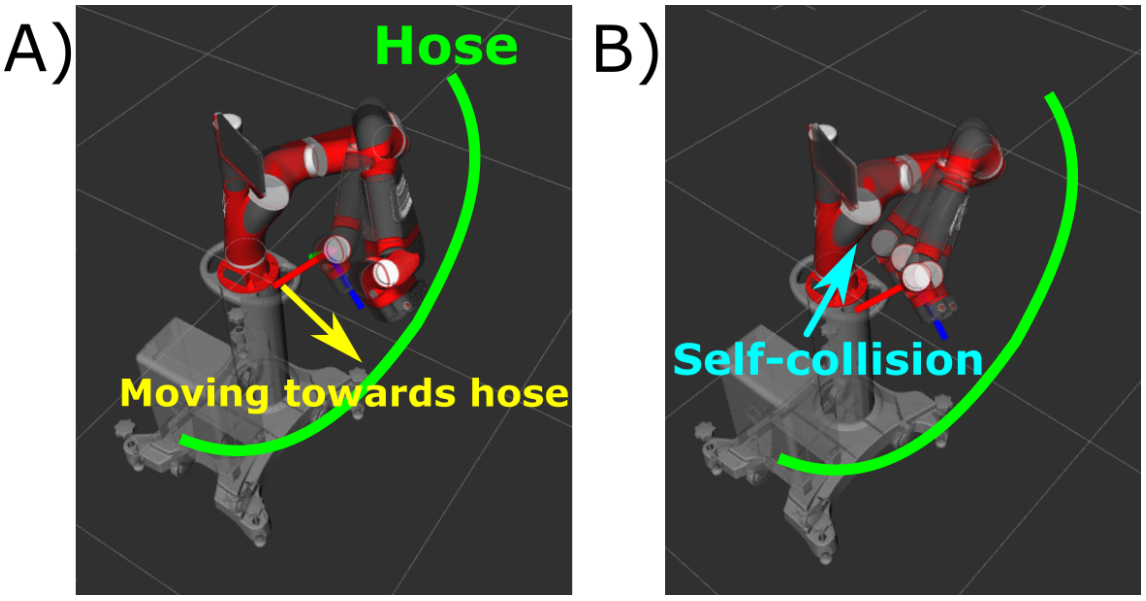


FIGURE 4.16: Comparing a successful task objective against a singularity driven self-collision. A) Simulation of the sawyer completing the task of moving towards the green target hose with a state-dependent polynomial constraint. B) Simulation of encountering a singularity, driving the system away from the task and causing a self-collision resulting in the failure of the task execution.

When estimating $\Lambda = (1, 0)$, which following Equation 4.11 selects the first row of $\mathbf{\Phi}$, $(0.905, 0.274)$ is learnt resulting in successful identification of the constraint.

Once learnt, the polynomial constraint is used following Equation 4.6 with $\boldsymbol{\pi}_{\mu}$ and $\boldsymbol{\pi}_{\tilde{\mu}}$ as a secondary control policy alongside the primary task extracted from the demonstrated data. Moreover, these are compared to using a point attractor and zero policy. The experiment is concluded with scores of 1.692, 1.696, 1.688 and 1.662 for μ, $\tilde{\mu}$, the point attractor and zero policy, respectively. Using $\tilde{\mu}$ results in the highest manipulability followed by μ, the point attractor and then the zero policy. To understand the differences between these scores, the paths of μ, $\tilde{\mu}$, the point

attractor and the zero policy are plotted for a visual comparison in Figure 4.17. As shown, μ and $\tilde{\mu}$ have overlapping paths, whereas other policies separate into different routes partway of the trajectory. Thus, the method is applicable in the real world for optimising a system's redundancy to have greater manipulability during a task.

As expected, the manipulability based cost function from Equation 4.6 maintains the highest average manipulability γ which is in line with the previous experiment.

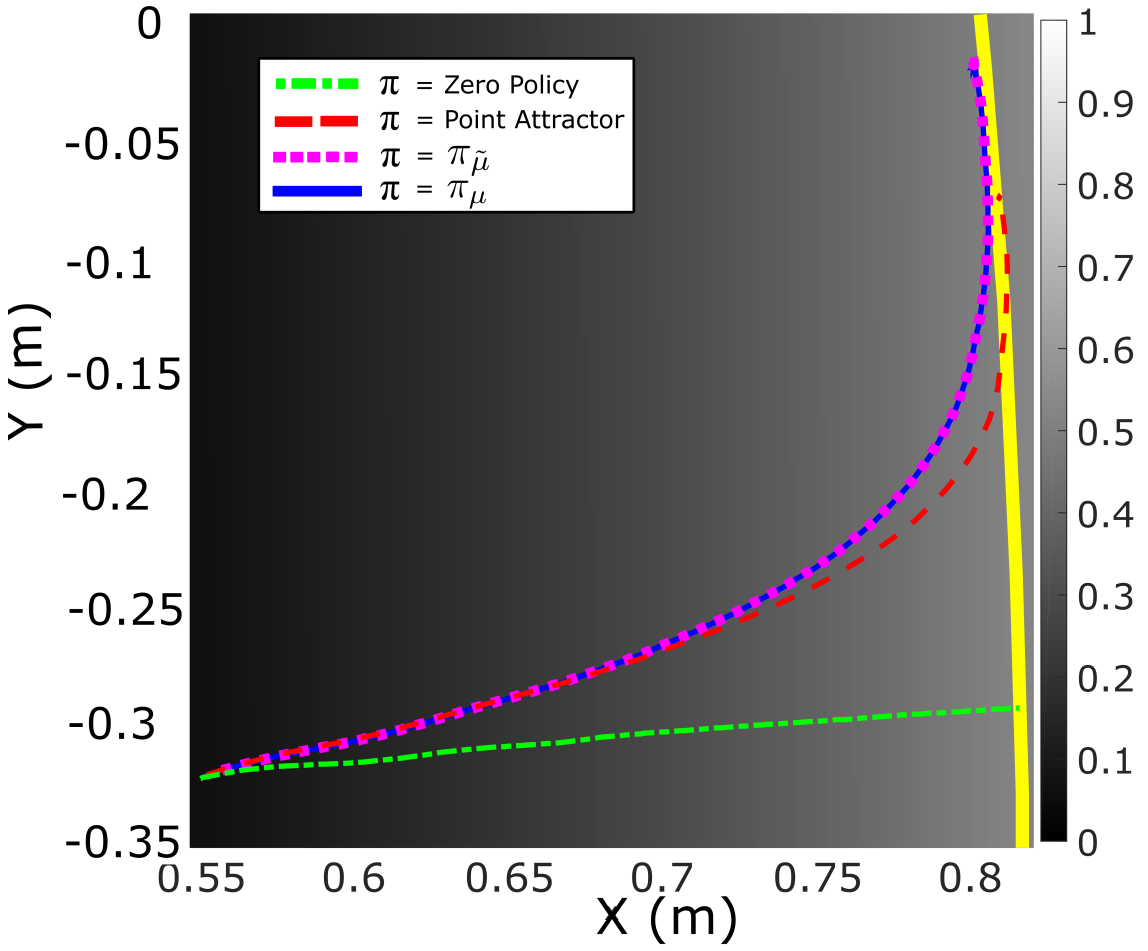


FIGURE 4.17: Resulting paths using different control policies

4.9 Evaluating Null Space controllers from Human Imitation

All sections until this point of the chapter primarily focus on kinaesthetic teaching through guided demonstrations. Therefore, this section, provides a brief look into comparing different null space control strategies when using demonstrations of a simulated human arm. These strategies are evaluated with respect to singularities in order to further demonstrate the importance of using manipulability optimisation to avoid undesirable behaviour.

4.9.1 Example: Reaching and Manipulating a Drawer

In this section, simulated demonstrations are gathered using a 3-link system modelled after the human arm consisting of the upper arm, lower arm and hand controlled by the wrist joint. These demonstrations follow ergonomic movements of the user and can be assessed by RULA (see Figure 2.9 further above). They can also be more suitably assessed by the reduced RULA (see Figure 5.4 further below). This is considered in more detail in the next chapter, with respect to a focus on the arm/wrist in a 2D lateral perspective without bending from the midline or twisting. Using the collected data, the trajectories are reproduced by a robotic system with a similar joint structure, (1) through the direct copying of the joint angles, (2) using the proposed approach which avoids singularity in the null space and (3) a zero-policy as a control to compare. The performance of the three reproductions are evaluated using the manipulability index score.

4.9.2 Experiment on Comparing Null Space Controllers

The aim of this evaluation is to assess the benefit of replacing mimicking of the ergonomic redundancy resolution when performed by a human, using the robot's manipulability-based control optimisation. While it is assumed that the controller using the manipulability-based approach should outperform the direct copying of the user in terms of avoiding singularities, a control, namely a zero-policy is included. This is to verify that improvements in the redundancy resolution are not simply due to a bad choice of the joint-space configuration of the user's arm as it comfortably performs the task. The setup is as follows.

Constrained motion data is gathered from a kinematic simulation of a 3-link planar robot with the ratio of the link lengths modelled after the average male's arm rounded to one decimal place [115]. The resulting system in which the link lengths (given as cm) are 33, 27 and 18 referring to the upper arm, lower arm and hand, respectively. The state and action space refer to the joint angle position and velocities, respectively, *i.e.,* $\mathbf{x} := \mathbf{q} \in \mathbb{R}^3$ and $\mathbf{u} := \dot{\mathbf{q}} \in \mathbb{R}^3$. The task space is described by the end-effector coordinates $\mathbf{r} = (r_x, r_y, r_\theta)^\top$, referring to the positions and orientation, respectively. With regards to the simulated demonstrator, start and end poses are limited to the human's range, this is detected when poses cannot result in a RULA score, meaning that it surpasses a person's joint limits. However, detection of motions past joint limits during the trajectory are not considered. The simulation runs at a rate of $66\,Hz$. Joint space motion of the system is recorded as it performs tasks while constrained to a target in the y-axis of end-effector space. A task constraint at state $\mathbf{x}$ is described through

$$\mathbf{A}(\mathbf{x}) = \mathbf{\Lambda}\mathbf{J}(\mathbf{x}) \tag{4.15}$$

where $\mathbf{J} \in \mathbb{R}^{3\times3}$ is the manipulator Jacobian, and $\mathbf{\Lambda} \in \mathbb{R}^{3\times3}$ is the selection matrix specifying the coordinates to be constrained. The following constraint is evaluated:

$$1) \quad \mathbf{\Lambda}_y = ((0,0,0),(0,1,0),(0,0,0))^\top.$$

This evaluation purely looks at comparing the performance of direct imitation against manipulability (subject to the task constraints) as well as a zero-policy used as a benchmark. Therefore, the true constraint model is used in the control policy $\boldsymbol{\pi}$; as successful singularity avoidance through the learnt constraint (by estimating the selection matrix) has already been established throughout this chapter. To simulate demonstrations of drawer opening/closing behaviour, the *Human* arm starts from a point chosen uniform-randomly $q_1 \sim U[-110°, 90°], q_2 \sim U[1°, 160°], q_3 \sim U[-40°, 40°]$ and moves to a task space target $\mathbf{r}^*$ following a linear point attractor policy

$$\mathbf{b}(\mathbf{x}) = \varpi(\mathbf{r}^* - \mathbf{r}) \tag{4.16}$$

where $\varpi = 5$ and $\mathbf{r}^*$ is drawn uniformly from $r^* \sim U[-2, 2]$. Targets without a valid RULA score (i.e. outside of a humans joint limits) are discarded. These trajectories by the human demonstrator are assumed to follow RULA optimisation (see Figure 2.9 further above) and more simply the reduced RULA (see Figure 5.4 further below). Looking at the control strategy, $\boldsymbol{\pi}$ selects the target state of the system by performing an exhaustive search on every possible combination of link poses to the nearest degree where the end-effector lies on the target point. It then selects the target state from the viable options based on which one has the lowest RULA score.

The selection process for the target state primarily enforces natural/ergonomic movement and also consistency, which makes it easier to separate the constraint from the control policy. For 5000 trials, each trial uses a uniform-randomly selected start pose and target (following the aforementioned criterion) repeated by three control policies (i) Direct imitation (of the demonstrator using ergonomic movements which are

represented as RULA) (ii) manipulability optimisation (using the true constraint model) and (iii) zero-policy. These result in three trajectories per trial using the aforementioned policies, thus a total of 15000 trajectories; of which there are 200 data points per individual demonstration in order to reach the target. A sample trajectory and evaluation of this experiment setup is shown in Figure 4.18 below.

The 5000 trials, consist of 5000 direct imitation, 5000 manipulability optimisation and 5000 zero-policy trajectories, which start at the same point and move towards the same target. Evaluating which system has the greatest average manipulability in each trial gives a score of 0, 184 and 4816 for the zero-policy, imitation and manipulability optimisation, respectively. This shows that the manipulability policy outperforms the other policies 96.32% of the time. Furthermore, when evaluating the manipulability at the final pose (once the target is reached at step 200) for all trials, the scores are 2, 39 and 4959 for the zero-policy, imitation and manipulability optimisation, thus our suggested policy is superior in 99.22% of all cases. The fewer cases where the zero-policy or direct imitation perform better are expected and can be attributed to the manipulability index being locally optimised, as opposed to globally. Thus, it does not guarantee the path with the greatest manipulability in all cases.

When evaluating singularity occurrences in the 5000 trials, where singularities in this case are determined by a minimum bound of 0.2 on the manipulability index, 165 occurrences are detected, i.e. 3.3%. Looking at the unique cases where only one of the three policies encounter a singularity in any given trial, the scores over all trials are 26, 67 and 5 for the zero-policy, imitation and manipulability, respectively. Moreover, delving further into those these particular cases, they occur when the system starts in a position which depending on the control policy forces it to be driven through a singular region before being able to reach the target. However when looking at these singular instances after 10% (0.6 seconds) of the trajectory

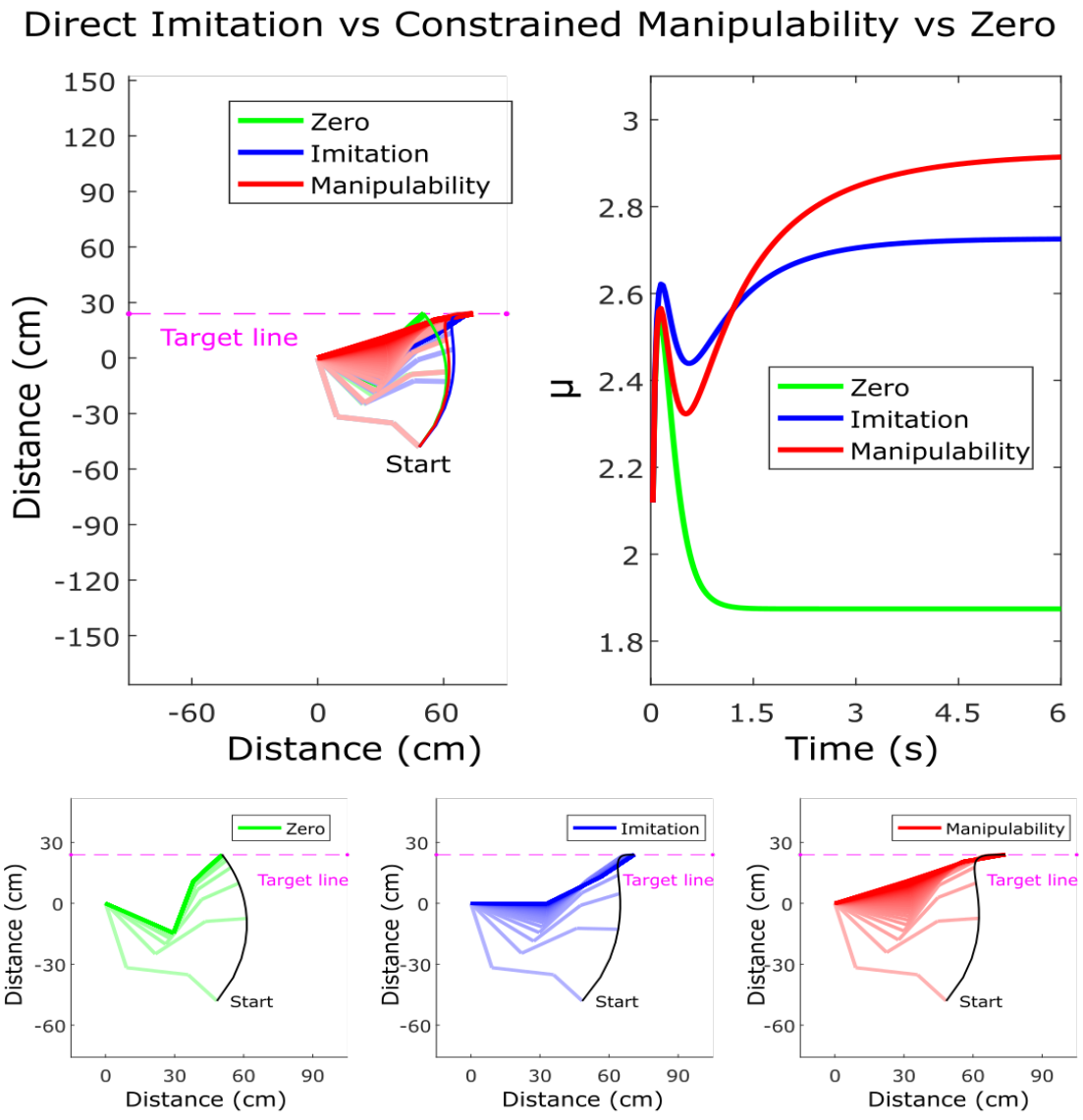


FIGURE 4.18: Comparing trajectories with different control policies from the data set. The top-left shows a stroboscopic plot with three trajectories which have the same start pose and task space target (pink dashed line), however each use a different control policy. It shows the zero-policy (green), direct imitation (blue) and the constrained manipulability (red) in $\boldsymbol{\pi}$ with their end-effector path which shows how the end-effector diverges. Link lengths given in cm are [33 27 18] for the robot's upper arm, lower arm and hand, respectively. Top-right shows the manipulability score for each policy over the entire trajectory, where the target is reached after 6 seconds. Bottom row displays each system individually and its taken path.

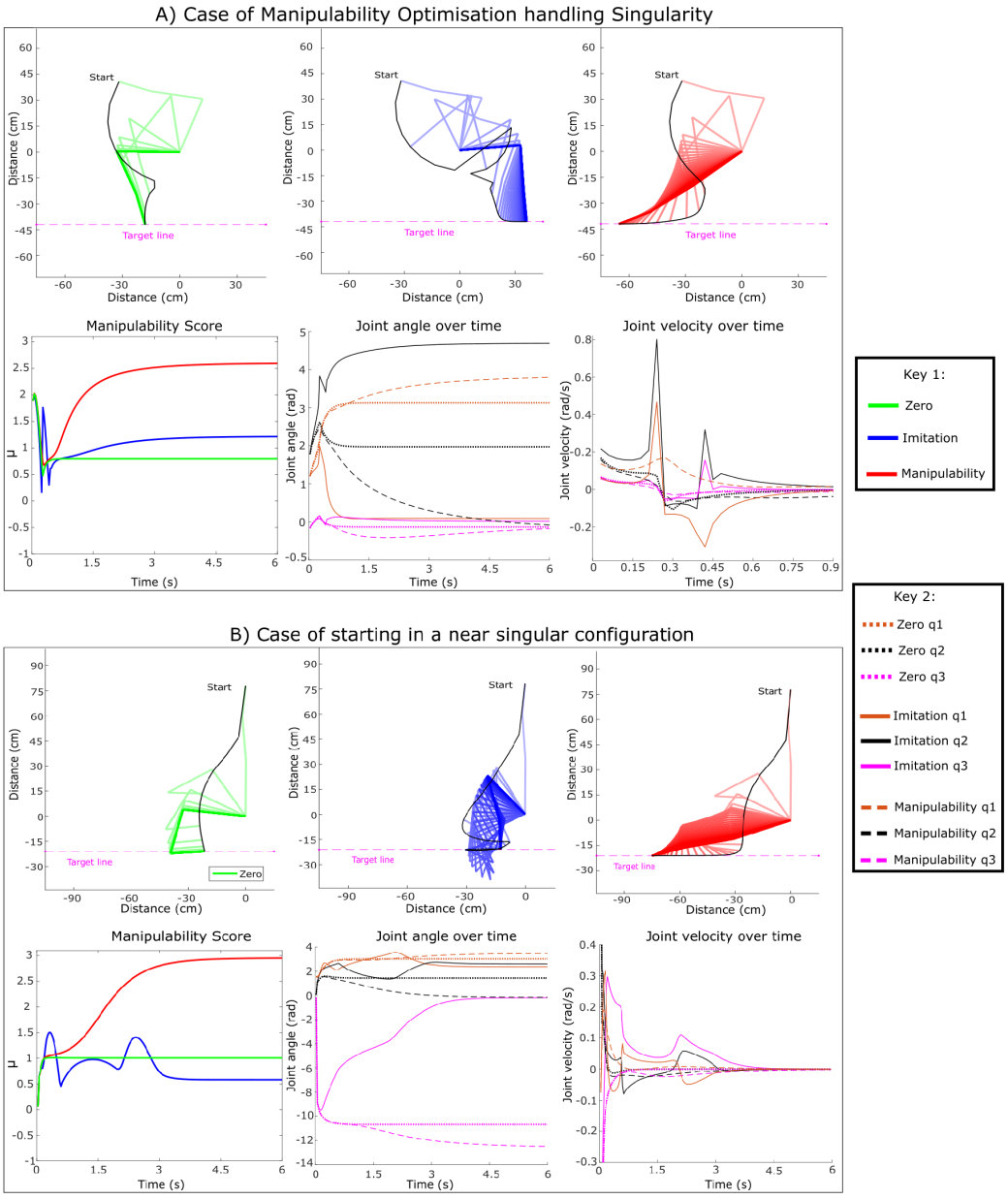


FIGURE 4.19: This figure shows how each policy behaves in near singular encounters. Two cases are presented where in both, a system has the same start pose and target, however under a different $\boldsymbol{\pi}$. In the top row of both cases, from left to right the policies used are the zero-policy (green), direct imitation (blue) and manipulability optimisation (red). In the bottom row of both cases, the 'manipulability score' uses the same aforementioned colour scheme. In the case of the 'joint angle over time' and the 'joint velocity over time', dotted lines refer to the zero policy, solid lines are the imitation policy and dashed lines are the manipulability-based policy. For each of the three links the colour scheme is q_1 =upper arm (orange), q_2 =lower arm (black) and q_3 = hand (pink).

is executed, the scores are 7, 12 and 0 for the zero-policy, imitation and manipulability, respectively. Only the manipulability optimisation manages to fully move away from all of the singular configurations. Looking at Figure 4.19 above, when encountering a near singular configuration, all three methods perform very differently. Figure 4.19-A shows the case whereby not optimising a system's manipulability will make it encounter an avoidable singularity. It also shows that by optimising the system according to its own *ergonomics* enables the task to be achieved following the smoothest trajectory. The benefit of optimising a system to its manipulability is not only in avoiding singular regions but also can be seen in how it handles near singular configurations when forced into such a position as shown in Figure 4.19-B. This case highlights when a near singular encounter is unavoidable, such as due to the starting pose. Optimisation using the system's manipulability makes it possible to move away from the singularity, such that it completes the task in a pose with a higher manipulability in comparison to the other policies. Moreover, although these other policies manage to reach the target (with various perturbations especially in the case of the imitation controller), they reach the target with a significantly lower manipulability increasing the risk of further encounters with singular configurations.

4.10 Conclusion

This chapter discusses using learning by demonstration to merge learning and manipulability-based control optimisation, so that it can be applied to a system allowing it to autonomously avoid singularities. This work considers the control of systems subject to uncertain constraints from a set of candidate constraints, due to the complexity and/or naivety of non-expert users. Through this approach, redundancy in the system is resolved through singularity avoidance which is learnt from the demonstrations. The taught system can perform task-oriented behaviour with a replaced and optimised null space control policy. The replaced policy can use the

learnt constrained manipulability as a learnt cost function for manipulability maximisation throughout the motion of the constrained system, not limited to kinematic systems.

First, results have been presented for a 3-link simulation in a 2D workspace and a real world experiment using the sawyer's arm in its 7DoF joint space, in which states and actions are given in the joint-space. All experiments are in agreement that manipulabilities can be learnt through demonstration. The simulation demonstrates using the learnt manipulability as a cost function to have the system avoid singularities while performing a task. It is important to note that the manipulability index is locally optimised in these experiments as opposed to globally and thus does not always guarantee the path with the greatest manipulability in all cases (see Section 4.5). The approach is verified in the real world using a robotic system with a high dimensional configuration space, showing that constraints can be learnt (through the unknown selection matrix) with enough precision to identify and avoid singular regions when substituting $\tilde{\mu}$ for μ and being used as a cost function. The optimised movements from the proposed approach result in an autonomous system that moves towards the goal while handling redundancy by moving away from singular regions through local manipulability optimisation. When compared to other control policies such as a zero policy and a point attractor, the proposed approach allows for the completion of the task, where the other policies are shown to succumb to the singularities within the same task, resulting in either no movement at all or unpredictable behaviour.

Another set of experiments are conducted, where results have been presented for end-effector motion data in a 2D simulation, 3D simulation and a real world experiment using the sawyer's arm. These experiments are also in agreement that unknown constraints can be learnt through demonstration. The simulation compares a polynomial and linear constraint in a 2D dimensional space, as well as, an additional circular constraint added when evaluating a 3D simulation. This addition

looks at how the approach performs in a setting of greater complexity in candidate constraints. It is shown that the selection matrix can be learnt to pick the correct constraint, such that the task manipulability index for different system configurations, within the support of the data, can be obtained. When compared to a linear point attractor and zero policy as the control policy, it is shown that the learnt manipulability based approach maintains the highest average manipulability. It works by moving towards the goal, while moving away from singular points through local optimisation. Moreover, it is also shown that the approach is robust even if the constraint isn't perfectly captured. This is because the overall direction towards manipulability maximisation can be ascertained (see Figure 4.11 and Figure 4.14 in Section 4.8).

Finally, further testing is performed on comparing different null space control strategies. It demonstrates the degree of improvement which can be expected when using manipulability optimisation over other policies. It does so by evaluating different aspects to encounters with singularities. It is shown that the proposed approach can outperform the other policies in maintaining the greatest average manipulability 96.32% of the time, as well as, have the highest manipulability for the final pose in 99.22% of all tested trials. Moreover, it is also shown how manipulability optimisation is able to quickly move away from singular regions where others struggle. This demonstrates its benefits given cases where singularities cannot be fully avoided, due to reasons such as a bad choice in the starting configuration.

Chapter 5

Exploiting Ergonomic Priors in Human-to-Robot Task Transfer

5.1 Introduction

In recent years, there has been a booming shift in the development of versatile, autonomous robots capable of performing increasingly complex tasks. Such robots are expected to enhance the capabilities of ordinary people to introduce automation into their lives by means of intuitively teaching robots task-oriented behaviour by demonstration [1, 116].

When humans perform task-oriented movement, it is often the case that there is a high level of redundancy, with the number of degrees of freedom (DoF) available to execute the task usually much higher than those required [25]. For instance, in the task of opening a drawer (see Figure 5.1 below), the primary objective is to *manipulate the drawer from the closed to the open position*, however, there is redundancy in the several possible ways this can be achieved. For example, a human performing this task can adopt different elbow postures, such as flaring it out at various degrees, while still managing to move the drawer (Figure 5.1-A below).

Having this flexibility is beneficial as it not only allows for multiple ways of achieving the task, but can also enhance efficiency or robustness, thereby enhancing overall performance.

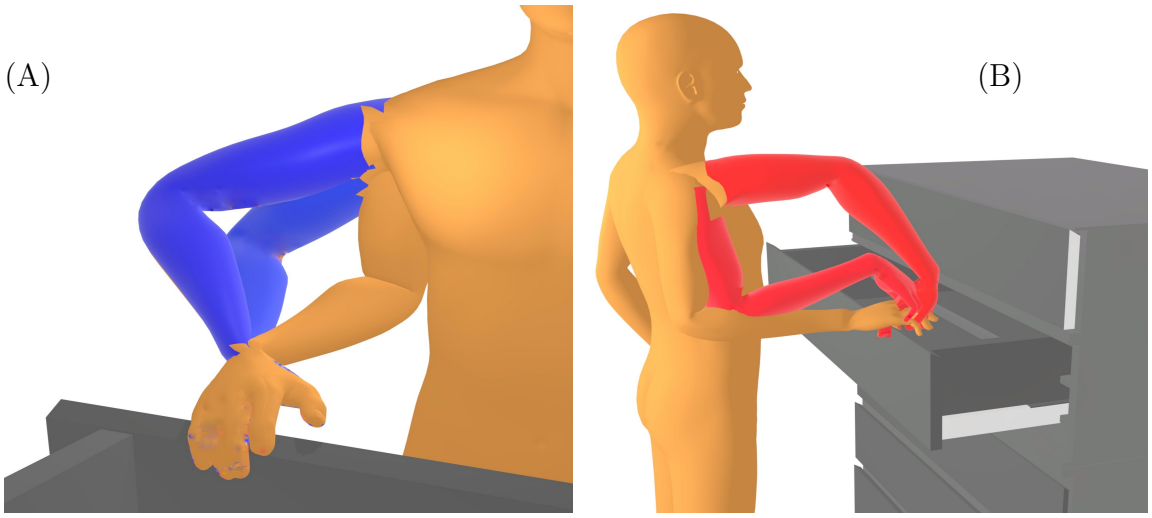


FIGURE 5.1: (A) Redundancy in elbow postures when opening a drawer. In absence of other constraints, humans tend to avoid using non-ergonomic postures (shown in blue). (B) A comfortable pose for a human is different to one that maximises manipulability in a robot with different joint limits.

Humans tend to take advantage of this flexibility in predictable and stereotypical ways, commonly by seeking to minimise discomfort or energy expenditure [117, 118]. For instance, despite the variety of postures that can be taken when a human opens a drawer, it is typical to keep ones wrist straight. This is done to avoid uncomfortable joint flexion. Furthermore, it is also typical to keep ones elbow down, to avoid working unnecessarily against gravity. Indeed, such features are codified in human ergonomics literature, to the point that they shape work environments and policy on safe working practice [87, 119, 120].

This flexibility is a hallmark of human behaviour that is an ongoing area of interest in research, which is not robustly captured by existing imitation learning approaches to an extent where it can be standardised outside of experimentation. Traditional imitation learning approaches tend to treat behaviours as monolithic control policies, and so do not lend themselves well to task-prioritised behaviours [1, 4, 5].

Similarly, a multi-DoF robotic system imitating the human can adopt different joint configurations that are consistent with maintaining the end-effector on the drawer handle, including those that closely match the human's posture. Therefore, the simplest way to have a robot learn is to match it to the human's posture as closely as possible when executing the task. However, such an approach neglects the differences in embodiment between human and robot that may lead to sub-optimal performance of the robot [2, 121]. For instance, maintaining a posture in which ones wrist joint is kept straight (Figure 5.1-B above), while comfortable for the human, may represent a singular posture for the robot that can lead to dangerous unstable movements. Moreover, for a robot with geared, non-backdriveable joints, it may cost little energy to maintain the elbow in a flared posture. Whereas moving the arm to a more human-like, elbow-down posture may actually expend energy unnecessarily. Such cases suggest that a more nuanced approach to human-to-robot behaviour transfer is required, that takes explicit account of the stereotypical features of human movement, and the desirability, or otherwise, to reproduce them in a robotic imitator.

To this end, this chapter investigates how stereotypical features of human demonstrators' posture control can be used to decompose observed behaviour into task-oriented and redundant components of motion. It highlights how humans have predictable ways of performing tasks which are captured in ergonomics. Moreover, it is shown how this predictability can be used to make assumptions on demonstration data to support learning of task constraints. Specifically, it presents a new method for programming by demonstration, whereby explicit use of the underlying null space control policy—as determined by the stereotypical or ergonomic features—is used. This learns the task and null spaces involved in the behaviour and their underlying constraints [13, 29]. The latter allows the original behaviour to be retargeted to an imitator robot that has a different kinematic embodiment. It does so by optimising movement according to the robot's own structure without causing any interference with the task goal [21]. Numerical and physical evaluations are reported in which

the proposed approach is applied to various learning problems. First, it is tested in a toy experiment to learn constraints without any prior knowledge regarding the constraints. The performance of the approach is tested on varying data lengths and noise. Then, a simulated 3DoF experiment is conducted where the system estimates a selection matrix to pick the constraints from a set of candidate constraints, which in this case is composed from rows of a Jacobian representing the human arm. Once its efficacy is established, it is followed by an evaluation which compares the approach to the state-of-the-art approach in [13, 29] (see Figure 5.6). Next, experiments are performed demonstrating the benefits of retargeting the system to resolve redundancy for obstacle avoidance (see Figure 5.7). This shows task reproduction from a demonstrator system to an imitator of a different embodiment. Finally, a real world experiment is shown, where demonstrations from a human are used to learn and reproduce the task space motions with a Sawyer, a 7DoF physical robot. This shows real world retargeting from a human to a robot of a different embodiment. The results indicate that learning in this way outperforms several competing methods, namely in [13, 23, 30, 31], in its ability to accurately learn the decomposition from relatively little data. This is supported by a comparative study where the constraint is learnt (using a selection matrix) from one trajectory of length 2 s (100 data points), with minimal assumptions made on the form of the data.

5.2 Background & Related Work

5.2.1 Human vs. Robot Optimisation

The promise of introducing collaborative robots for automation outside traditional manufacturing settings is their usability by novice users, *i.e.,* those potentially with domain-expertise but little knowledge of robotic engineering. It is envisaged that

such users would teach task-oriented skills through demonstration, thereby avoiding the need for formal training in the techniques and concepts familiar to roboticists.

With this in mind, there is a need to take account of the natural motor behaviour of novices when designing interfaces and approaches to the programming by demonstration of such systems. It has long been established, for example, that *differences in embodiment* of the human musculoskeletal system and most robotic actuation systems mean that direct imitation of human motor behaviour is suboptimal for robots [2]. This suggests the need for *selectivity in imitation*, whereby only task-critical features of behaviour are mimicked, while secondary features–those that are idiosyncratic to the demonstrator–can be neglected. This has been shown to be effective, for example, by Zhao et al., where detrimental features of demonstrator reaching profiles, arising from coupled impedance parameters, were effectively removed in robotic reproductions of the task.

One major source where idiosyncrasy in human behaviour arises, comes down to people's natural propensity to seek comfort and minimise fatigue in movement, subject to any applicable task constraints. When a task is demonstrated by a novice user, they will typically adopt postures that, for instance, avoid working against gravity or limb flexion/extension away from what's comfortable. These tendencies are also reinforced by the design of working environments that are typically set up with (human) comfort in mind.

However, while such ergonomic considerations can promote efficiency in human work, they can be deleterious to performance when reproducing such behaviour in robots. For example, lower back-drivability or high friction in robotic joints, can make it inefficient to constantly seek postures mimicking the demonstrator (*e.g.,* elbows and hands hanging down, see Figure 5.2) when there is no energetic need to do so.

With these issues in mind, this paper explores the ergonomic differences between

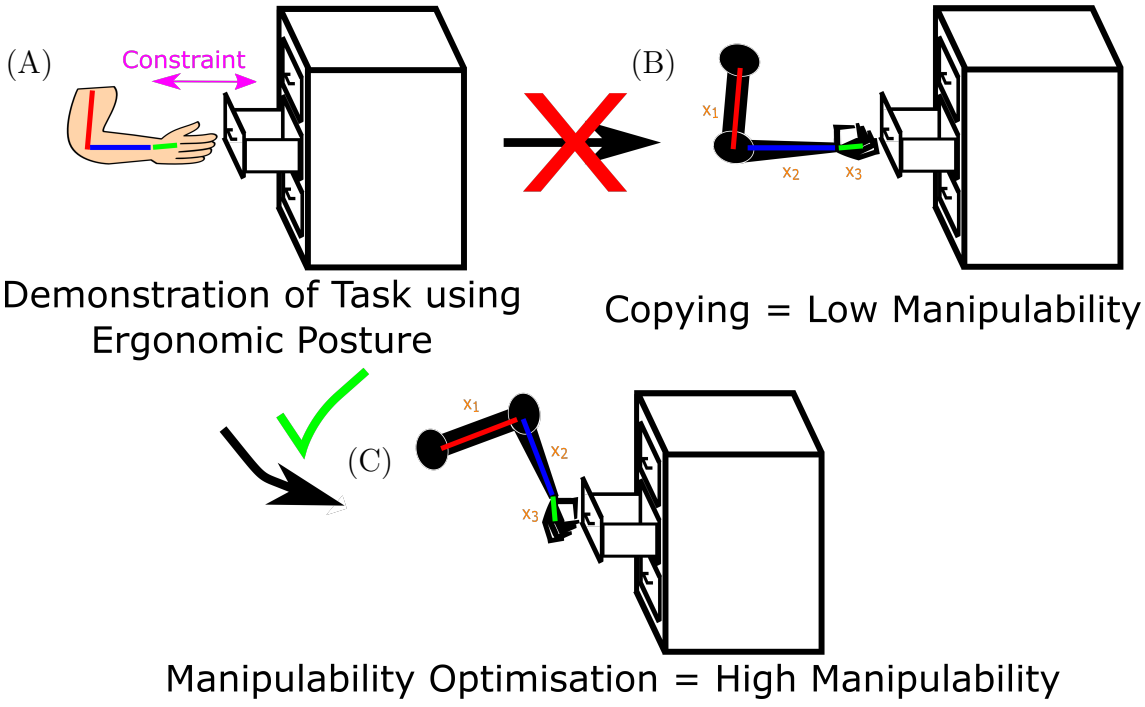


FIGURE 5.2: Illustrative flow chart showing direct imitation compared to adapting behaviours to be more "robot ergonomic". (A) Human-drawer opening may be imitated either by (B) directly copying the sequence of joint postures, or (C) optimising the redundant parts of the movement for the robot (here, by maximising the system's manipulability).

humans and robots, with a view to examining the extent to which removal of human-ergonomic factors in demonstrations can enable better task performance in robot reproductions of behaviour.

5.2.2 Human-ergonomic Demonstrations

In most scenarios where ordinary people are asked to provide natural demonstrations of a given task, in the absence of external constraints, it can generally be assumed the behaviour observed will be optimised for efficiency and comfort according to their embodiment. This means that postures adopted by the user in the course of a demonstration are likely to show stereotypical traits that promote these goals. For example, when reaching for a target such as a drawer handle, typically people will adopt a low-energy posture with little limb flexion, thereby minimising the

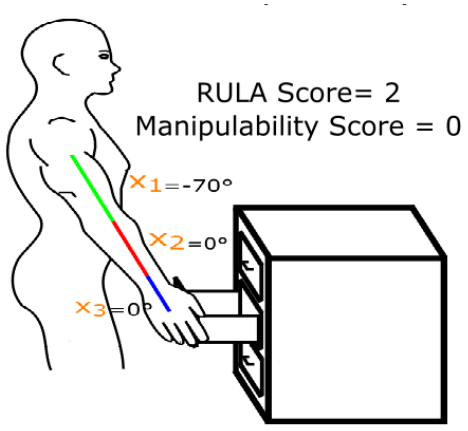


FIGURE 5.3: Typical drawer opening posture that is ergonomic for humans but not for robots. According to RULA, the posture shown has a relatively good score of 2 (upper arm position $= -70° \rightarrow$ upper arm score 1, lower arm position $= 0° \rightarrow$ lower-arm score 2 and wrist $= 0° \rightarrow$ wrist score 1, see Figure 5.4). However, for inverse kinematic control in a robotic system, this pose is singular (manipulability score 0), and therefore unlikely to meet the task demands.

effort needed to counter gravity (see Figure 5.3). This is further encouraged by the arrangement of objects in human work-spaces, that are also designed to maximally promote the comfort of their occupants. In the aforementioned example, the location of the drawer handle and its path during opening are configured to minimise the need for deviation from this posture.

Note that, these stereotypical features are *secondary to the task*—that is, they will tend to be promoted to seek comfort and minimise fatigue in movement—but are *subject to any applicable task constraints*. This means that they can be inhibited if the task demands it. For example, in drawer opening, the default rest posture of the shoulder is not maintained, since the hand must be lifted to the drawer handle for the task. Furthermore, if maintaining the elbow-down posture during opening *conflicts with the task* (*e.g.,* would result in a collision with an obstacle), then the task space extends to the elbow elevation and overrides the default behaviour.

Such interactions of humans with their environment have long been studied in the field of *human ergonomics* [87, 119], with a view to improving and optimising the design of workspaces. As part of this, a number of measures exist that aim to quantify good design and working practice which assess various exposure factors such as posture, force and movement frequency, to name a few [51]. In monitoring posture for manual tasks, for example, a popular system is the Rapid Upper Limb Assessment

(RULA) [57]. It works by scoring static poses of individual joints of the upper limb, where an overall lower score implies that the posture is more ergonomic. It also assesses the effect of repeated actions. For reference, a reduced version of the RULA worksheet adapted from that of ErgonomicsPlus© is shown in Figure 5.4 below. Commonly, ergonomic assessments consider force, however this reduced version is applicable to tasks where 1) static postures are not held for over 10 minutes, 2) tasks are not repeated over four times per minute and 3) held loads are below 2kg [57]. Assuming these conditions are met, there is no negative impact on the overall score from the muscle and force components and these can therefore be omitted. Several other measures can also be used to categorically assess risk in static poses such as the Loading on the Upper Body Assessment (LUBA) or New Ergonomic Posture Assessment (NERPA), however studies show that out of these RULA performs best in assessing the risk factor of postures that incite muscle fatigue and discomfort which can lead to work-related musculoskeletal disorders [82].

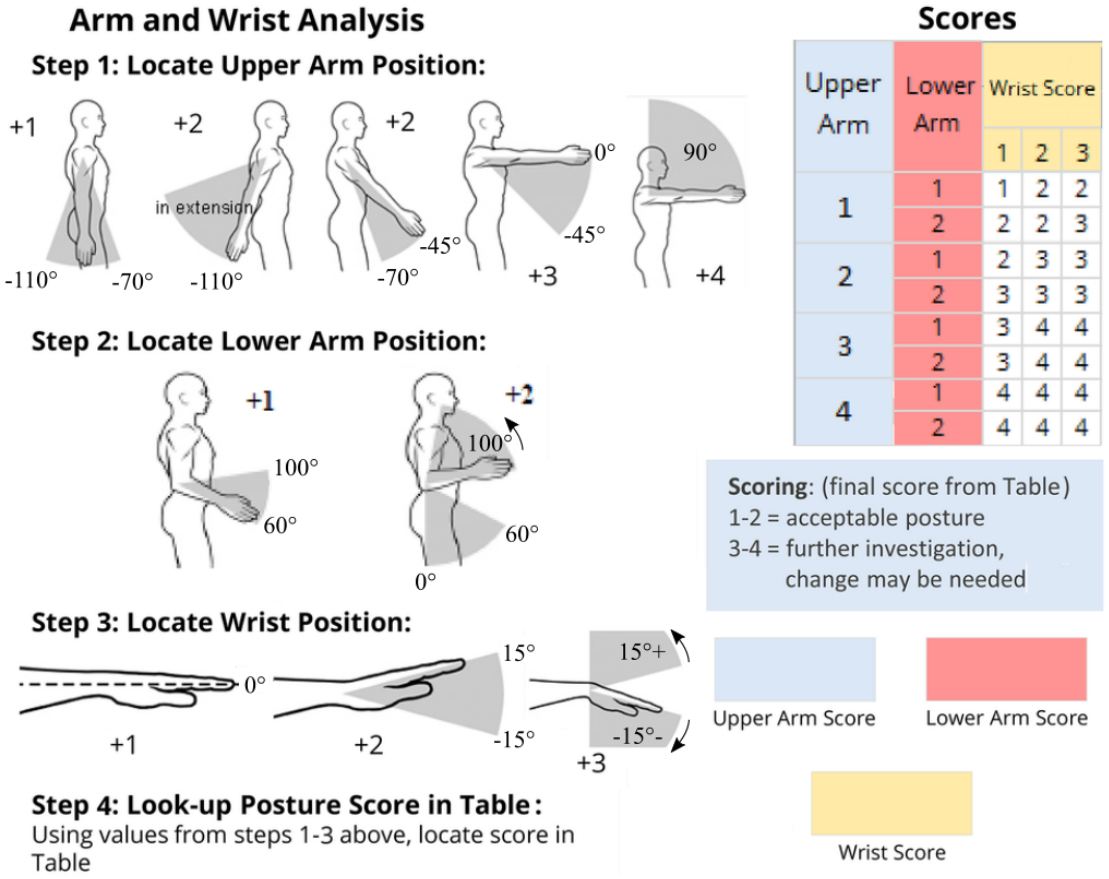


FIGURE 5.4: Reduced version of the Rapid Upper Limb Assessment (RULA) worksheet which focuses the ergonomics assessment to a 2D lateral perspective (arm/wrist analysis only without bending from the midline or twisting). This is adapted from [120]. To compute the score for any static pose, follow steps 1-3 successively so that three individual scores are obtained for the upper arm, lower arm and wrist, respectively. Then, enter those individual scores into the table 'Scores' (on the right), where the upper arm and lower arm are used in turn to select the appropriate row, and the wrist score selects the correct column. This results in a final score, which can be checked against the 'Scoring' box below the table, to verify whether it is an acceptable posture or may require change. See Figure 5.3 for an example on a static posture.

5.2.3 Robot Imitation of Ergonomic Behaviours

While working practices and environments that promote ergonomic postures can lead to greater efficiency in humans, the same cannot be said for robotic systems tasked with entering those same environments to perform similar tasks. It has long

been established that the differences in embodiment between humans and robots, including the kinematic structure [122] and dynamic [2], mean that direct imitation of human behaviour is suboptimal.

In the context of postural imitation of human ergonomic behaviour, this issue is particularly pronounced. For instance, considering the drawer opening example, while the posture shown in Figure 5.3 above scores well in terms of human ergonomics (RULA score: 2 points, considered an acceptable posture), the same is not true for a robot imitator. In this case, because the arm is fully extended, this posture represents a singular pose for standard robotic inverse kinematics controllers. This can lead to unpredictable and potentially dangerous behaviour. Conversely, a pose in which the upper arm is flexed to be parallel with the horizontal would be a high-stress pose for a human (and therefore score proportionately worse by RULA). However, may not be a difficult posture to maintain for a robot with low back-drivability, since its joints can effectively be locked in place with no energy cost.

It is evident that, when imitating ergonomic human behaviour in robotic systems, direct imitation may be unsuitable in many cases. This chapter focuses on work in programming by demonstration with naive users. The work aims to take advantage of assumptions regarding how humans have predictable preferences for resolving redundancy to some extent. This can be based on their similar embodiment as well as being inclined towards seeking postures that maximise comfort. With this in mind, constraint learning can be performed on a reduced range of poses the demonstrator is most likely to opt for when performing a task. A method is proposed based on this assumption (shown in Section 5.3). Moreover, this research aims to establish a link when learning, to enable behaviours to be ergonomic for both the human teacher and robotic learner, while still meeting the demands of the task.

5.2.4 Task Prioritised Behaviour

To better capture this task-prioritised view of behaviour, several studies have recently focused on modelling demonstrations hierarchically (see Section 2.2.2-2.2.3 for details). In these, movement is decomposed into the *task space*—the DoF required for the primary task—and a *null space* (*i.e.,* the remaining DoF). This draws on several well-established hierarchical control schemes, such as Liegeois' redundant kinemetic control scheme [123], or Khatib's Operational Space Formulation [124].

In this view, actions $\mathbf{u} \in \mathbb{R}^{\mathcal{Q}}$ are assumed to take the form of Equation 2.3. $\mathbf{A}(\mathbf{x}) \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ is a matrix describing a system of $\mathcal{S}$-dimensional constraints following Equation 2.2.

In this view, $\mathbf{b}(\mathbf{x}) \in \mathbb{R}^{\mathcal{S}}$ represents the *task space policy* describing the primary task to be accomplished, and the lumped term $\mathbf{v} = \mathbf{A}^{\dagger}(\mathbf{x})\mathbf{b}(\mathbf{x})$ represents that policy projected into the configuration space. $\boldsymbol{\pi}(\mathbf{x})$ represents the *null space policy*, that encapsulates any actions in the configuration space secondary to the task. Note that, it is typically the case that $\mathbf{b}$ is unknown (since this is the task that should be learnt by demonstration), and $\mathbf{A}$ (and therefore $\mathbf{N}$) is also not explicitly known (since this describes the space in which the unknown task is defined).

The key insight of this chapter is that in many cases, *prior knowledge of* $\boldsymbol{\pi}$ *may be assumed*, since it commonly represents the *stereotypical features of secondary movements*. Furthermore, as shown in Section 5.3, knowledge of $\boldsymbol{\pi}$ enables efficient estimation of the other quantities in Equation 2.3 ($\mathbf{v}$ and $\mathbf{N}$). These can in turn be used to separate out the task-oriented part of the demonstrations, and thereby replace the secondary components with a control policy tailored to the imitator's embodiment.

5.2.5 Learning the Decomposition

Several prior studies have examined the possibility of robot learning by demonstration using the representation Equation 2.3-2.4, and in particular the possibility of learning $\mathbf{v}$ or $\mathbf{A}$ under the assumption that only $\mathbf{x}$ and $\mathbf{u}$ are observable. However, as noted in Section 5.2.2, in many cases such an assumption is *overly stringent* and can result in degraded estimation performance.

Depending on the assumptions made on its representation (see Section 5.3.3), one of several learning methods can be used to estimate $\mathbf{A}$ [13, 23, 30]. However, all of these methods rely on the ability to separate the lumped task space term $\mathbf{v}$ (or, equivalently, the null space term $\mathbf{w}$) from the demonstrations. The issue here is that learning performance is highly dependent on the quality of the separation. These studies rely on the same approach, first proposed by Towell *et al.* [29], which uses variations in the task space policy $\mathbf{b}$ and consistency in the null space policy $\boldsymbol{\pi}$ to form an estimate of the separation. This has several limitations in practice making it not straightforward for the separation to work. First, it can be difficult to ensure that the data is 'rich' enough in terms of the variations seen in the task space policy $\mathbf{b}$. Second, to learn the null space component, if working with data which contains different several distinct null space control policies, it is important to separate the data into subgroups and learn within each subset individually. However, this diminishes the learning quality as it tends to make less data available within each subgroup. These requirements can hamper the method's efficacy as increasingly complex systems and constraints are considered. The approach proposed here does not have such prerequisites—instead, it exploits prior knowledge of the control policy $\boldsymbol{\pi}$, a component that can often be estimated through an understanding of stereotypical behaviour or consideration of human ergonomics.

5.3 Method

In this section, a new method is defined for estimating the null space projection matrix $\mathbf{N}$ in redundant systems where some prior knowledge of the redundancy resolution strategy is assumed available.

5.3.1 Data

The proposed method works on data given as $\mathcal{N}$ pairs of observed states $\mathbf{x}_n$ and actions $\mathbf{u}_n$ collected from task-oriented movement demonstrations. It is assumed that (i) observations are in the form presented in Equation 2.3, (ii) $\mathbf{A}$, $\mathbf{N}$ and $\mathbf{b}$ are not explicitly known for any given observation and (iii) $\boldsymbol{\pi}$ is *known* (or a good estimate is available).

As noted in Section 5.2, assumption (iii) is reasonable depending on several factors, including the task at hand and the environment. In most circumstances, healthy human demonstrators will tend to perform tasks in a way that *promotes comfort*. In the experiments reported here, this tendency is captured by assuming that the secondary control policy is a point attractor

$$\boldsymbol{\pi}(\mathbf{x}) = \boldsymbol{L}(\mathbf{x}^* - \mathbf{x}) \tag{5.1}$$

where $\boldsymbol{L}$ is a gain matrix and the point of attraction $\mathbf{x}^*$ is a posture that scores highly according to standard ergonomic assessment procedures such as RULA [120]. While many ergonomic measures are applicable depending on the experiment, RULA is selected as it is quick and simple to classify static postures by looking at joint angles. Moreover, it focuses on the upper body which is in line with the drawer handling experiments, and provides a constant joint posture range for optimal scoring [57]. It is important to note that experiments consist of reaching tasks conducted from a standing posture. The topic of environmental constraints is briefly discussed in

Section 2.2.2 , however this works does not consider these. Including environmental exploits would alter what is considered the *most ergonomic* posture identified by RULA and would require different ergonomic measures. Moreover, while many ergonomic measures also consider energy/force, in the case of RULA these can be omitted if certain conditions are met as they do not impact the score in any way (see Section 5.2.2).

5.3.2 Learning the Null Space Projection Matrix

The proposed method works by exploiting the orthogonality between the task and null space parts in Equation 2.3. Specifically, noting that $\mathbf{v}^\top \mathbf{w} = \mathbf{w}^\top \mathbf{v} = 0$, Equation 2.3 can be written

$$\mathbf{w}^\top \mathbf{u} = \mathbf{w}^\top \mathbf{v} + \mathbf{w}^\top \mathbf{w} = \mathbf{w}^\top \mathbf{w} \tag{5.2}$$

yielding the identity

$$\mathbf{w}^\top (\mathbf{u} - \mathbf{w}) = 0. \tag{5.3}$$

An estimate of $\mathbf{N}$, and therefore the null space component $\mathbf{w}$, can be formed by choosing $\tilde{\mathbf{w}} = \tilde{\mathbf{N}}\boldsymbol{\pi}$ consistent with this identity by minimising[1]

$$E[\tilde{\mathbf{N}}] = \sum_{n=1}^{\mathcal{N}} ||\boldsymbol{\pi}_n^\top \tilde{\mathbf{N}}_n (\mathbf{u}_n - \boldsymbol{\pi}_n)||. \tag{5.4}$$

This minimisation problem can be solved using various non-linear optimisation tools. In this case Matlab's `fmincon` is used with the interior-point algorithm, which is a non-linear optimisation solver for constrained multivariable functions.

[1]For brevity, here, and throughout the chapter, the notation $\mathbf{a}_n$ is used to denote the quantity $\mathbf{a}$ evaluated on the nth sample. For example, if $\mathbf{a}$ is a vector quantity computed from the state $\mathbf{x}$, then $\mathbf{a}_n = \mathbf{a}(\mathbf{x}_n)$.

5.3.3 Representation of the Constraints

In order to efficiently learn $\tilde{\mathbf{N}}$, a suitable representation needs to be selected. The approach chosen here follows that first proposed by Lin *et al.* [13, 23], which represents $\tilde{\mathbf{N}}$ in terms of an underlying constraint matrix $\tilde{\mathbf{A}}$ according to Equation 2.4. This has been shown to be effective both for unstructured problems (*i.e.,* where the form of the constraint matrix $\mathbf{A}$ is completely unknown) and for situations where some features (*i.e.,* candidate rows) of $\mathbf{A}$ are available.

5.3.3.1 Unit Vector Representation of A

Following [23], if the form of $\mathbf{A}$ is completely unknown, it can be represented using a (potentially, state-dependent) set of $\mathcal{S}$ orthonormal vectors

$$\tilde{\mathbf{A}} = \left[\boldsymbol{\alpha}_1^\top \, \boldsymbol{\alpha}_2^\top \, \cdots \, \boldsymbol{\alpha}_{\mathcal{S}}^\top\right]^\top \tag{5.5}$$

where $\boldsymbol{\alpha}_s = (a_{s,1}, a_{s,2}, ..., a_{s,\mathcal{Q}})$ corresponds to the sth constraint in the observations. The latter can be constructed iteratively by selecting vectors orthonormal to one another where the sth vector has the form

$$\begin{aligned}
a_{s,1} &= \cos\theta_1 \\
a_{s,2} &= \sin\theta_1 \cos\theta_2 \\
a_{s,3} &= \sin\theta_1 \sin\theta_2 \cos\theta_3 \\
&\vdots \\
a_{s,\mathcal{Q}-1} &= \prod_{\nu=1}^{\mathcal{Q}-2} \sin\theta_\nu \cos\theta_{\mathcal{Q}-1} \\
a_{s,\mathcal{Q}} &= \prod_{\nu=1}^{\mathcal{Q}-1} \sin\theta_\nu .
\end{aligned} \tag{5.6}$$

The resulting matrix is represented by a total of $\mathcal{P} = \mathcal{S}(2\mathcal{Q} - \mathcal{S} - 1)/2$ parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_{\mathcal{P}})^\top$.

5.3.3.2 Representation of A with Candidate Rows

In the case that prior information about the form of the constraint matrix is available, this can be incorporated into the estimate using the approach first proposed by [13]. Here, a suitable representation of the constraint matrix is

$$\tilde{\mathbf{A}} = \tilde{\mathbf{\Lambda}}\mathbf{\Phi} \tag{5.7}$$

where $\tilde{\mathbf{\Lambda}} \in \mathbb{R}^{\mathcal{S}\times\mathcal{P}}$ is a *selection matrix* (to be estimated during learning) and $\mathbf{\Phi} \in \mathbb{R}^{\mathcal{P}\times\mathcal{Q}}$ is a (possibly, state-dependent) feature matrix. The rows of the latter can take generic forms such as a series of polynomials, or can contain candidate constraints if there is prior knowledge of those which are potentially affecting the system. For instance, one may choose $\mathbf{\Phi} = \mathbf{J}(\mathbf{x})$, the Jacobian of the manipulator, so that $\tilde{\mathbf{A}} = \tilde{\mathbf{\Lambda}}\mathbf{J}(\mathbf{x})$ encodes constraints on the motion of specific degrees of freedom in the end-effector space.

5.3.4 Estimating the Components of the Behaviour

Once $\tilde{\mathbf{N}}$ is estimated, the decomposition of the behaviour into task-oriented and null space parts are straightforward. The null space component is given as

$$\tilde{\mathbf{w}} = \tilde{\mathbf{N}}\boldsymbol{\pi} \tag{5.8}$$

and the task space part is computed as

$$\tilde{\mathbf{v}} = \mathbf{u} - \tilde{\mathbf{w}}. \tag{5.9}$$

Note that, given the estimate $\tilde{\mathbf{A}}$, $\tilde{\mathbf{N}}$ can be computed using Equation 2.4, and an estimate of the task space policy $\tilde{\mathbf{b}}$ can be obtained using Equation 2.2.

5.3.5 Substituting the non-task oriented Behaviour

As noted in Section 5.1, in many cases the redundancy resolution strategy seen in demonstrators' task-oriented behaviour may be ill-suited to the robot imitator. The proposed method provides a simple means of *retargeting the behaviour to the robotic system*, while maintaining the task-oriented parts. Specifically, this is achieved by replacing the controls from Equation 2.3 with

$$\mathbf{u} = \tilde{\mathbf{A}}^{\dagger}\tilde{\mathbf{b}} + \tilde{\mathbf{N}}\boldsymbol{\psi} \tag{5.10}$$

where $\boldsymbol{\psi}$ is a (possibly, state-dependent) redundancy resolution policy for the robot. For instance, $\boldsymbol{\psi}$ could be chosen so as to avoid robot-specific joint limits or singularities [21]. Alternatively, if the task space trajectory is predictable, it can be used in combination with global optimisation in the null space [114].

5.4 Evaluation

In this section, the proposed approach is first examined through a toy experiment without any knowledge of the constraints. Then, using candidate constraints represented as a Jacobian, learning with a more complex 3-link planar system is tested. On top of evaluating performance, comparisons to the state-of-the-art [13, 29] are conducted. In addition to this, demonstrations of retargeting are presented, first, for obstacle avoidance using a system's null space component to resolve redundancy differently, and second, for retargeting from one system to another of a different

embodiment. After this, the performance of the approach in the context of programming by demonstration with a human demonstrator and a 7DoF physical robot is evaluated.[2]

5.4.1 Toy Problem

The aim of the first evaluation is to test the robustness of the proposed method using data from a simple, two-dimensional system with a one-dimensional task space. The setup (based on [13]) is as follows.

Constrained motion data is gathered from a two-dimensional system with a one-dimensional constraint $\mathbf{A} = \boldsymbol{\alpha} \in \mathbb{R}^{1\times 2}$. Movement in the task space is defined by the constraint matrix and occurs in the direction of the unit vector $\hat{\boldsymbol{\alpha}} = (\cos\theta, \sin\theta)$. This direction is selected from a uniform-random distribution $\theta \sim U[0°, 180°]$ at the start of each trial. The task space policy is a linear point attractor $b(\mathbf{x})_i = r^* - r, i \in \{1\}$, where r is the position in the task space and r^* is the target point. To simulate varying tasks, the task space targets are selected randomly $r^* \sim U[-2, 2]$ for each trial.

In the below, learning performance is reported for three different secondary control policies $\boldsymbol{\pi}$, namely,

1. *A linear policy*: $\boldsymbol{\pi}(\mathbf{x}) = -\mathbf{L}\bar{\mathbf{x}}$ where $\bar{\mathbf{x}} \coloneqq (\mathbf{x}^\top, 1)^\top)$ and $\mathbf{L} = ((2, 4, 0)^\top, (1, 3, -1)^\top)^\top)$.
2. *A limit cycle*: $\dot{\rho} = \rho(\rho_0 - \rho^2)$ with radius $\rho_0 = 0.75\,m$, angular velocity $\phi = 1\,rad/s$, where ρ and ϕ are the polar representation of the state, *i.e.,* $\mathbf{x} = (\rho\cos\phi, \rho\sin\phi)^\top$.
3. *A non-linear (sinusoidal) policy*:
   $\boldsymbol{\pi} = (\cos z_1 \cos z_2, -\sin z_1 \sin z_2)^\top$ where $z_1 = \pi x_1$ and $z_2 = \pi(x_2 + \frac{1}{2})$.

[2]The data supporting this research are openly available from King's College London at `http://doi.org/[link to be made available on acceptance]`. Further information about the data and conditions of access can be found by emailing `research.data@kcl.ac.uk`.

TABLE 5.1: Test data NMSE in $\tilde{\mathbf{w}}$ (mean±s.d.)$\times 10^{-15}$ and $E_{\tilde{\mathbf{N}}}$ (mean±s.d.)$\times 10^{-8}$ over 50 trials for different $\boldsymbol{\pi}$.

$\boldsymbol{\pi}$	$E_{\tilde{\mathbf{w}}}$	$E_{\tilde{\mathbf{N}}}$
Linear	1.2147 ± 2.6458	0.4263 ± 1.3396
Limit-cycle	0.5462 ± 0.7043	1.1616 ± 1.1682
Sinusoidal	0.3020 ± 0.5741	1.6685 ± 1.9316

The training data consists of 150 data points, drawn uniform randomly across the space $(\mathbf{x})_i \sim U[-1, 1], i \in \{1, 2\}$. For testing, a further 150 data points are used, generated through the same procedure as above. The constraint is learnt by finding a $\boldsymbol{\theta}$ which minimises Equation 5.4. In each trial, performance is measured using two metrics. First, the normalised mean squared error (NMSE) in the estimated null space component is evaluated[3]

$$E_{\tilde{\mathbf{w}}} = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} ||(\mathbf{w}_n - \tilde{\mathbf{w}}_n) \oslash \boldsymbol{\sigma}_{\mathbf{u}}||^2. \tag{5.11}$$

where $\boldsymbol{\sigma}_{\mathbf{u}} \in \mathbb{R}^{\mathcal{Q}}$ is a vector containing the element-wise standard deviation of the observations $\mathbf{u}$. Note that, as $\mathbf{w}_n = \mathbf{N}_n \boldsymbol{\pi}_n$ and $\tilde{\mathbf{w}}_n = \tilde{\mathbf{N}}_n \boldsymbol{\pi}_n$, this measure is equal to the NPPE (defined in Appendix A.1) [23].

Second, the normalised error measure (Equation 5.4) is evaluated

$$E_{\tilde{\mathbf{N}}} = \frac{1}{\mathcal{N}||\sigma_{\mathbf{u}}||^2} \sum_{n=1}^{\mathcal{N}} ||\boldsymbol{\pi}_n^\top \tilde{\mathbf{N}}_n (\mathbf{u}_n - \boldsymbol{\pi}_n)||. \tag{5.12}$$

It indicates the performance of the minimisation function using only known information from given data and is therefore applicable for practical applications.[4] The experiment is repeated 50 times.

[3]The notation $\mathbf{C} = \mathbf{A} \oslash \mathbf{B}$ denotes *Hadamard* (element-wise) *division* of $\mathbf{A}$ by $\mathbf{B}$, *i.e.,* $(\mathbf{C})_{ij} = (\mathbf{A})_{ij}/(\mathbf{B})_{ij}$.

[4]To evaluate the fitness of $\tilde{\mathbf{v}}$, noting that $\mathbf{v} + \mathbf{w} = \tilde{\mathbf{v}} + \tilde{\mathbf{w}}$ can be written as $\mathbf{v} - \tilde{\mathbf{v}} = -\mathbf{w} + \tilde{\mathbf{w}}$ returns the identity $\mathbf{v} - \tilde{\mathbf{v}} = -(\mathbf{w} - \tilde{\mathbf{w}})$, where the error in both components are opposites. Thus, results for the fitness of $\tilde{\mathbf{w}}$ can also be considered the same for $\tilde{\mathbf{v}}$ and therefore $\tilde{\mathbf{v}}$ is omitted.

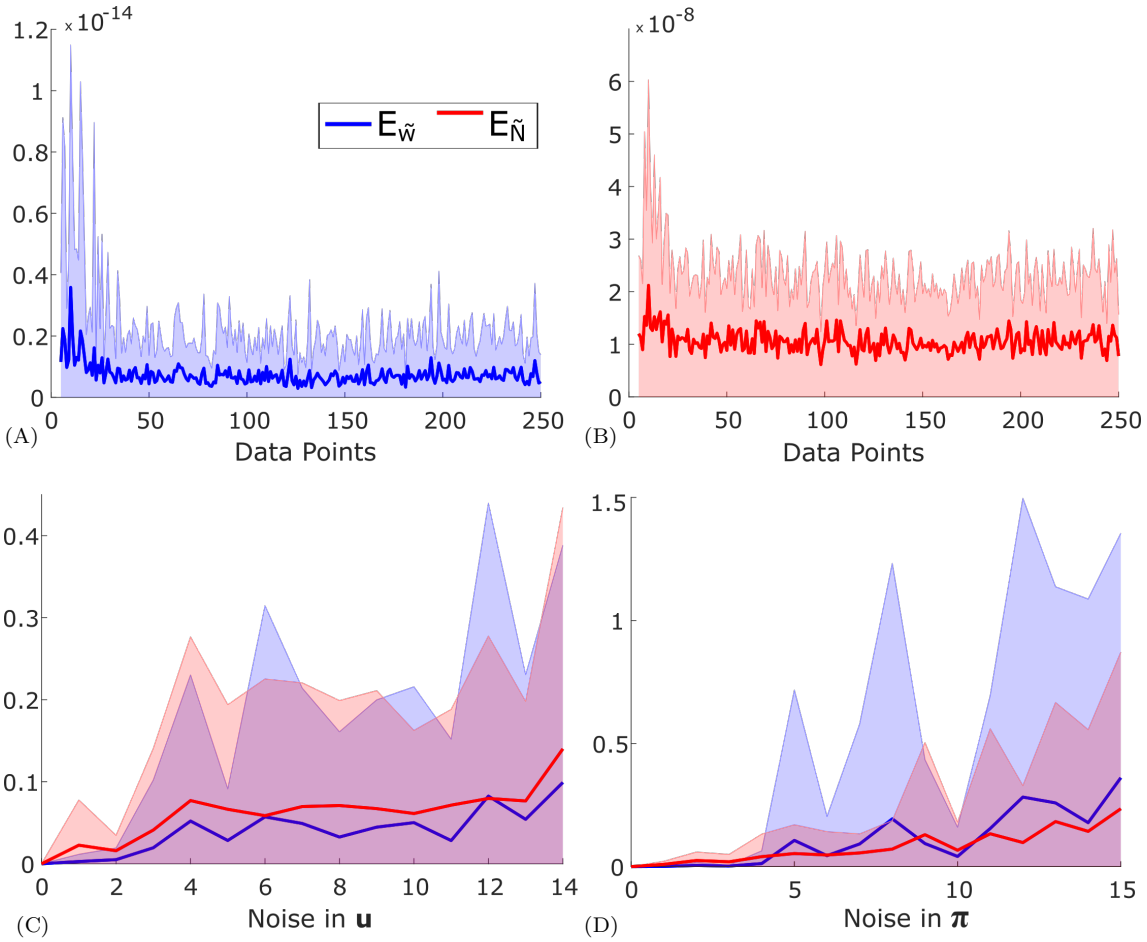


FIGURE 5.5: NMSE in $\tilde{\mathbf{w}}$ and $E_{\tilde{\mathbf{N}}}$ (mean±s.d. over 50 trials) for (A) increasing number of data points for $E_{\tilde{\mathbf{w}}}$, (B) increasing number of data point for $E_{\tilde{\mathbf{N}}}$, (C) increasing noise levels in $\mathbf{u}$ and (D) increasing noise levels in $\boldsymbol{\pi}$. Mean results are plotted as thick lines and their respective standard deviation are the shaded areas of a similar lighter tone.

The NMSE in $\tilde{\mathbf{w}}$ and $E_{\tilde{\mathbf{N}}}$ are presented in Table 5.1. As can be seen, $\tilde{\mathbf{N}}$ is successfully estimated with errors in $\tilde{\mathbf{w}}$ less than 10^{-14} and $\tilde{\mathbf{N}}$ with less than 10^{-7} for all of the policies considered. These low errors shows that the constraint matrix can be estimated with very high precision if knowledge of $\boldsymbol{\pi}$ is available. The overall performance of $\tilde{\mathbf{w}}$ is seen to be very roughly around twice as accurate as $E_{\tilde{\mathbf{N}}}$. This shows that the task and null space components can generally be reproduced with greater accuracy than indicated by just evaluating $E_{\tilde{\mathbf{N}}}$.

To further characterise the performance of the proposed approach, the experiment is repeated with (i) data sets of varying sizes ($5 < \mathcal{N} < 250$), (ii) varying levels

of noise in the training data $\mathbf{u}_n$ represented as $N(0, \epsilon\sigma_{\mathbf{u}}^2)$ additive white Gaussian noise where $0 < \epsilon < 0.14$ and (iii) varying levels of noise in the estimated $\boldsymbol{\pi}_n$ where $0 < \epsilon < 0.15$ for 50 trials using the limit cycle policy. The latter case simulates error in the assumed $\boldsymbol{\pi}$ and thereby allows for evaluation of the proposed approach in face of an inaccurate estimate of the true underlying redundancy resolution strategy. The results are plotted in Figure 5.5 above.

As shown in Figure 5.5-A, the NMSE in $\tilde{\mathbf{w}}$ is less than 10^{-14} for both mean and standard deviation with as few as five data points. As the number of data points increases, so does the accuracy for minimising $\tilde{\mathbf{w}}$. The performance of the method seems to plateau after around 25 data points. This shows that the approach can learn constraints with as few as five data points and for optimal performance with at least around 25 data points. Figure 5.5-B shows errors of less than 10^{-7} for both mean and standard deviation in $\tilde{\mathbf{N}}$. It follows a similar trend to the previous evaluation with respect to accurate learning with as few as five data points and optimal performance after at least around 25 data points. It can also be observed that the learning performance is very roughly half compared to $E_{\tilde{\mathbf{w}}}$ which was also observed in Table 5.1. Looking at Figure 5.5-C above, there is a clear trend with a degrading mean accuracy and greater standard deviation as the noise in $\mathbf{u}$ increases. The mean error in $\tilde{\mathbf{w}}$ stays below 0.1 when $\epsilon <= 0.14$ and for mean error in $\tilde{\mathbf{N}}$ when $\epsilon <= 0.13$. It can also been seen that the error in $\tilde{\mathbf{N}}$ is greater compared to $\tilde{\mathbf{w}}$ in most cases which is in agreement with prior experiments. Looking at Figure 5.5-D above, the accuracy decreases, with greater standard deviation, as the error in the assumed $\boldsymbol{\pi}$ increases. The mean error in $\tilde{\mathbf{w}}$ stays below 0.1 when $\epsilon < 0.05$, however when $\epsilon <= 0.1$ only 2 mean values are shown to produce an error above 0.1. The mean $E_{\tilde{\mathbf{N}}}$ stays below 0.1 when $\epsilon <= 0.08$ and only has a single instance where the mean value is above 0.1 where $\epsilon <= 0.1$. Comparing $E_{\tilde{\mathbf{w}}}$ and $E_{\tilde{\mathbf{N}}}$, while both have similar mean performance, the standard deviation of $E_{\tilde{\mathbf{N}}}$ is noticeably smaller. This is expected, as $E_{\tilde{\mathbf{N}}}$ relies on knowledge of the noisy estimate of $\boldsymbol{\pi}$ to obtain $\tilde{\mathbf{N}}$,

whereas $E_{\tilde{\mathbf{w}}}$ compares this to the true $\boldsymbol{\pi}$ to present the error within the estimated null space component.

5.4.2 Simulated Three Link Planar Arm

The aim of the next evaluation is to test the performance of the proposed method on a more complex system with non-linear constraints which simulates a real world system more accurately. The setup is as follows.

Constrained motion data is gathered from a kinematic simulation of a three-link planar robot with uniform links of length $10\,cm$. The state and action space refer to the joint angle position and velocities, respectively, *i.e.,* $\mathbf{x} := \mathbf{q} \in \mathbb{R}^3$ and $\mathbf{u} := \dot{\mathbf{q}} \in \mathbb{R}^3$. The task space is described by the coordinates $\mathbf{r} = (r_x, r_y, r_\theta)^\top$ referring to the end-effector positions and orientation, respectively. The simulation runs at a rate of $50\,Hz$.

Joint space motion of the system is recorded as it performs tasks under different constraints in the end-effector space. Specifically, a task constraint at state $\mathbf{x}$ is described through

$$\mathbf{A}(\mathbf{x}) = \boldsymbol{\Lambda}\mathbf{J}(\mathbf{x}) \tag{5.13}$$

where $\mathbf{J} \in \mathbb{R}^{3\times3}$ is the manipulator Jacobian assumed known *a priori*, and $\boldsymbol{\Lambda} \in \mathbb{R}^{3\times3}$ is a diagonal selection matrix (to be estimated), with elements $\boldsymbol{\lambda} = (\lambda_x, \lambda_y, \lambda_\theta)^\top$ along the diagonal, indicating which coordinates should be included in ($\lambda_i = 1$) or excluded from ($\lambda_i = 0$) the task space. In the results reported below, the following six selection matrices are considered: (i) $\boldsymbol{\Lambda}_x$ where $\boldsymbol{\lambda} = (1, 0, 0)^\top$, (ii) $\boldsymbol{\Lambda}_y$ where $\boldsymbol{\lambda} = (0, 1, 0)^\top$, (iii) $\boldsymbol{\Lambda}_\theta$ where $\boldsymbol{\lambda} = (0, 0, 1)^\top$, (iv) $\boldsymbol{\Lambda}_{x,y}$ where $\boldsymbol{\lambda} = (1, 1, 0)^\top$, (v) $\boldsymbol{\Lambda}_{x,\theta}$ where $\boldsymbol{\lambda} = (1, 0, 1)^\top$, and (vi) $\boldsymbol{\Lambda}_{x,\theta}$ where $\boldsymbol{\lambda} = (0, 1, 1)^\top$.

To simulate demonstrations of reaching behaviour, the robot end-effector starts from a point chosen uniform-randomly $q_1 \sim U[0°, 10°], q_2 \sim U[90°, 100°], q_3 \sim U[0°, 10°]$ to

TABLE 5.2: Mean±s.d. $E_{\tilde{\mathbf{w}}}$ and $E_{\tilde{\mathbf{N}}}$ on testing data for different null space policies over 50 trials. The figures for $E_{\tilde{\mathbf{w}}}$ are (mean±s.d.)$\times 10^{-11}$ and for $E_{\tilde{\mathbf{N}}}$ are (mean±s.d.)$\times 10^{-7}$.

$\boldsymbol{\pi}$	$E_{\tilde{\mathbf{w}}}$	$E_{\tilde{\mathbf{N}}}$
$\boldsymbol{\Lambda}_x$	0.0741 ± 0.0291	1.1527 ± 2.3654
$\boldsymbol{\Lambda}_y$	12.1467 ± 18.6578	11.1195 ± 9.1619
$\boldsymbol{\Lambda}_\theta$	0.1496 ± 0.3732	0.2445 ± 0.1700
$\boldsymbol{\Lambda}_{x,y}$	5.6114 ± 10.3401	5.1969 ± 6.5849
$\boldsymbol{\Lambda}_{x,\theta}$	0.0139 ± 0.0320	0.8522 ± 1.0982
$\boldsymbol{\Lambda}_{y,\theta}$	0.0476 ± 0.1357	0.6719 ± 0.6424

a task space target $\mathbf{r}^*$ following a linear point attractor policy

$$\mathbf{b}(\mathbf{x}) = \mathbf{r}^* - \mathbf{r} \tag{5.14}$$

where $\mathbf{r}^*$ is drawn uniformly from $r_x^* \sim U[-1, 1]$, $r_y^* \sim U[0, 2]$, $r_\theta^* \sim U[0°, 180°]$. As the secondary control policy, a simple point attractor of the form

$$\boldsymbol{\pi}(\mathbf{x}) = \boldsymbol{L}(\mathbf{x}^* - \mathbf{x}) \tag{5.15}$$

is used, where $\mathbf{x}^*$ is arbitrarily chosen as $x_1 = 10°$, $x_2 = -10°$, $x_3 = 10°$and $\boldsymbol{L} = 1$. For each of the cases (i)–(vi) above, 100 trajectories are generated, each containing 50 data points. 50% of the samples are provided for learning and the remainder reserved as unseen testing data. Finally, this whole experiment is repeated 50 times.

The NMSE in $\tilde{\mathbf{w}}$ and $E_{\tilde{\mathbf{N}}}$ on the testing data are presented in Table 5.2 above. As shown, the constraints are successfully learnt with $E_{\tilde{\mathbf{w}}}$ less than 10^{-9} and $E_{\tilde{\mathbf{N}}}$ less than 10^{-5} in all cases. The $E_{\tilde{\mathbf{w}}}$ is roughly half of the $E_{\tilde{\mathbf{N}}}$ which is in agreement with previous experiments. Overall, the constraint matrix can be accurately estimated using data from the observed demonstrations and knowledge of the control policy, without having to explicitly know how the constraints affect the system's motions.

To further evaluate the performance of the proposed method, it is compared to

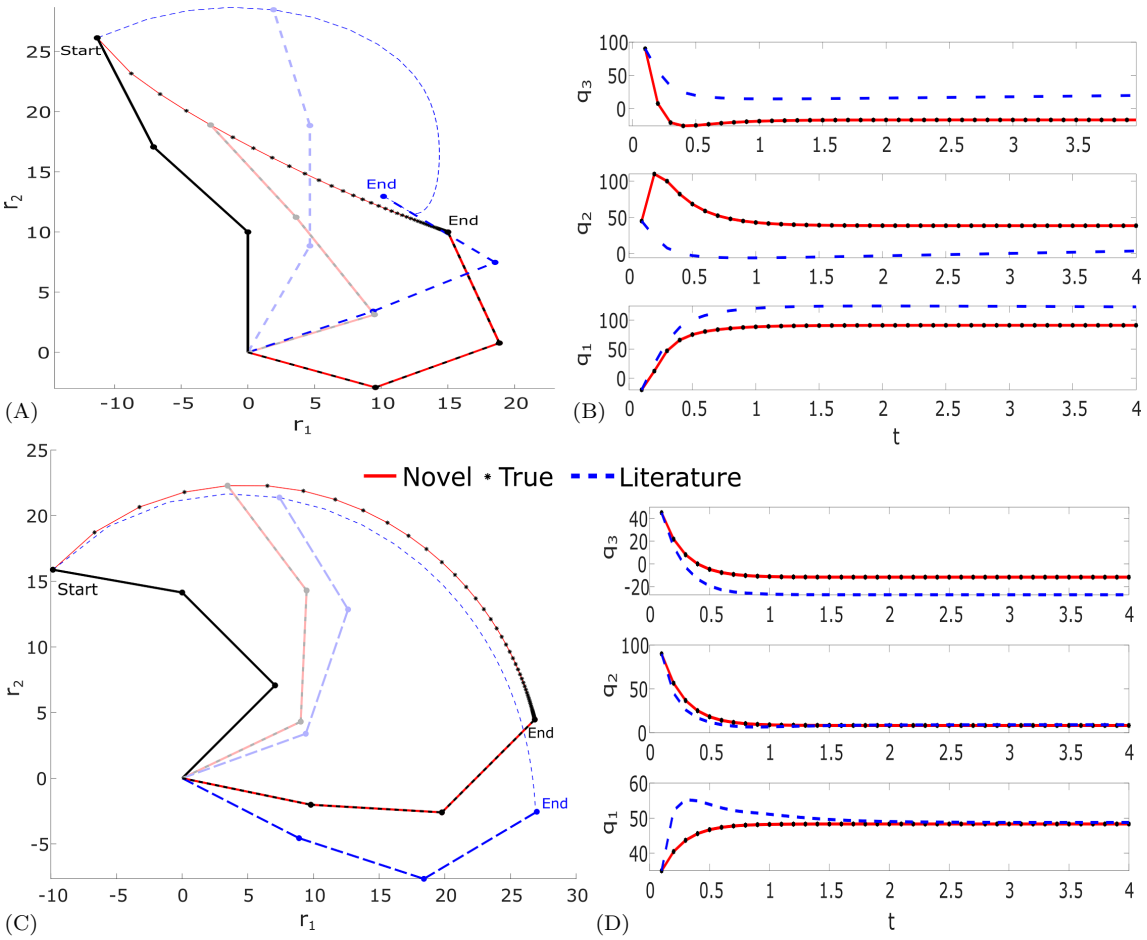


FIGURE 5.6: Reproducing the ground-truth movement (dotted-black) in both learnt task and null space using the proposed method (solid-red), and the state-of-the-art method [29] (dashed-blue) to learn the null space component and constraint $\mathbf{A}$ to obtain $\mathbf{b}$ [13]. (A) Arm visualisation for example task under constraint space $\mathbf{r} = (x, y)$, (B) Joint angle positions during example movement in task space $\mathbf{r} = (x, y)$, (C) Arm visualisation for example task under constraint space $\mathbf{r} = \theta$ and (D) Joint angle positions during example movement in task space $\mathbf{r} = \theta$.

the current state-of-the-art in [13, 29]. As discussed in Section 5.2.5, while there are many applicable methods to learn the constraint matrix with acceptable performance including [13], they all rely on the method proposed in [29] for separation of the observed actions. Following [29], Figure 5.6 shows an example of using a learnt constraint to generate a new trajectory. In this experiment, the new trajectory is reproduced using $\tilde{\mathbf{A}}$ which is learnt from a separate training data set, $\mathbf{u}$, $\mathbf{x}$ and $\boldsymbol{\pi}$, where the latter three are given. Firstly, training data consists of one trajectory of

length $2\,s$ (100 data points) with a random constraint in x, y. Using the state-of-the-art approach in [29], $\mathbf{u}$ is separated into the task and null space components. Now that the null space component is learnt, it is used with $\mathbf{u}$, $\mathbf{x}$ and the approach in [13] to obtain $\tilde{\mathbf{A}}$. On the other hand, the novel approach uses $\mathbf{u}$, $\mathbf{x}$ and $\boldsymbol{\pi}$ to directly obtain $\tilde{\mathbf{A}}$ (without having to separate the task and null space components). Now that both approaches have resulted in a learnt constraint, a ground-truth test trajectory is produced subject to the same true constraints in x, y of the training data. Its start pose is $q_1 = 90°, q_2 = 45°, q_3 = -20°$ and the x, y position of the end-effector moves towards the target point $(15, 10)^\top$ which reaches convergence in 4 seconds. To compare the novel and literature approach, both use $\mathbf{x}$ of this ground-truth data to start at the same position. Both produce $\tilde{\mathbf{v}}$ with their respectively learnt $\tilde{\mathbf{A}}$ and use this with $\mathbf{u}$ to estimate $\tilde{\mathbf{b}}$ following Equation 2.2. The literature approach already has $\tilde{\mathbf{w}}$ which was obtained using [29]. The novel approach uses Equation 2.4 to obtain $\tilde{\mathbf{N}}$ and uses the known $\boldsymbol{\pi}$ to produce $\tilde{\mathbf{w}}$. Both approaches use this information to iteratively reproduce the ground-truth data and similarly run for 4 seconds which is shown in Figure 5.6-A and Figure 5.6-B. This experiment is repeated for a constraint in θ shown in Figure 5.6-C and Figure 5.6-D.

As can be seen in Figure 5.6-A and Figure 5.6-B above, the proposed approach is shown alongside the true policy as well as the current state-of-the-art [29] for comparison. The new method follows the true joint trajectories accurately. The state-of-the-art approach on the other hand takes a different route in both the end effector trajectory as well as joint space leading to a different target. In Figure 5.6-C and Figure 5.6-D above, the task space target is set as the orientation of the end-effector which moves towards the target angle of 45°. The novel approach accurately reproduces the movements under this 1DoF constraint unlike the state-of-the-art method comprised of [29] and [13].

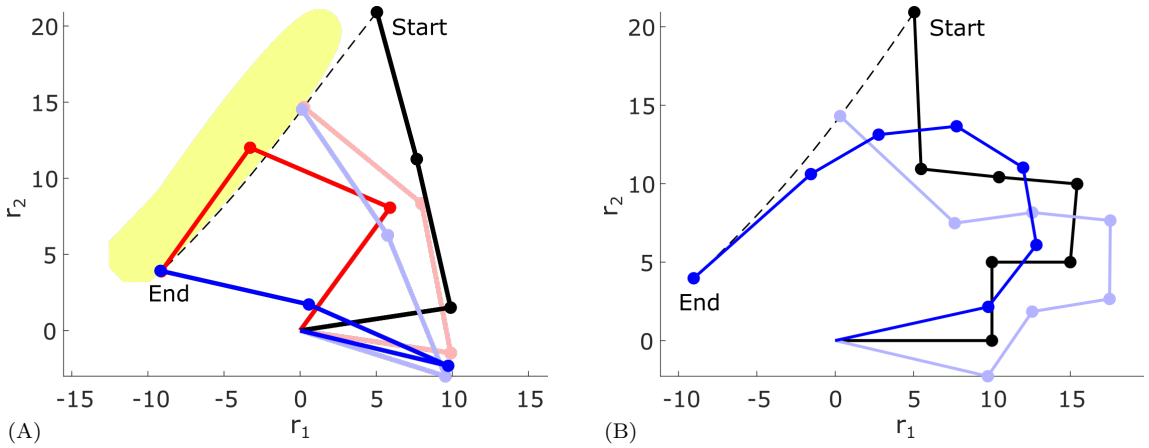


FIGURE 5.7: Retargeting behaviour with an imitator robot (A) with the same embodiment as the demonstrator but located near to an obstacle, and (B) with a different kinematic structure. The demonstrated movement is illustrated in red, while that of the imitators is in blue. The yellow region indicates an obstacle.

As mentioned in Section 5.3.5, the proposed approach allows *retargeting* of task-oriented behaviour by substituting the demonstrator's redundancy resolution strategy with one better suited to the robot. More concretely, consider the scenario where it is desired to reproduce a demonstrated reaching movement (i) with a robot with identical embodiment to the demonstrator, but is located right next to an obstacle (such that there is the risk of collision), see Figure 5.7-A, and (ii) with a robot that has a different kinematic structure to the demonstrator (different number and length of links). In the following, the feasibility of retargeting to these scenarios are assessed. Starting with the reaching movement to be reproduced, a typical trajectory is taken from the training data (given in the absence of any obstacles) described above. Specifically, the example chosen uses $\mathbf{\Lambda} = \mathbf{\Lambda}_{x,y}$, $\mathbf{w}$ is derived from the policy Equation 5.15, $\mathbf{r}^* = (-9.12, 3.89)^\top$ and $\mathbf{q} = (8.67°, 94.18°, -2.32°)^\top$. This movement is *retargeted* by using the learnt $\tilde{\mathbf{A}}$ to derive $\tilde{\mathbf{b}}$, and then applying Equation 5.10 with a replacement null space control policy.

In the case of the robot located next to an obstacle, retargeting simply consists of selecting an appropriate null-space control policy $\boldsymbol{\psi}$. Here, $\boldsymbol{\psi}(\mathbf{x}) = \boldsymbol{L}_r(\mathbf{x}_r^* - \mathbf{x})$ is used, where $\boldsymbol{L}_r = 5$ and $\mathbf{x}_r^* = (-320°, 100°, 50°)^\top$. The resulting movements *with*

and *without* retargeting are shown in Figure 5.7-A above (blue and red figures, respectively). As can be seen, if the arm directly imitated the demonstration (red figure) a collision would occur (second and third links overlap with the yellow region). This would not only jeopardise the success of the task but also potentially cause damage to the system. In contrast, starting at the same start point, the retargeted controller (blue figure) successfully completes the task (as it converges to the same target point in end-effector space). Moreover, by resolving its redundancy differently, it avoids the collision.

In the case of the robot with the different kinematic structure, retargeting is achieved as follows. As noted above, the constraint in this system is represented in the form Equation 4.3 where the feature matrix is selected as the Jacobian of the demonstrator's embodiment (*i.e.,* $\mathbf{\Phi} = \mathbf{J}$) and the selection matrix $\tilde{\mathbf{\Lambda}}$ is learnt. Since the rows of $\mathbf{\Phi}$ represent meaningful quantities (here, the relationship between the joint space and the end-effector position and orientation), a correspondence is drawn between these and the equivalent quantities for the new arm (*e.g.,* if the first row of $\mathbf{J}$ relates to the Jacobian for the r_x coordinate, the corresponding row of the Jacobian $\mathbf{J}_r$ for the imitator is selected), and $\mathbf{A}_r$ is constructed accordingly. Substituting this into Equation 5.10 gives the controller

$$\mathbf{u} = \mathbf{A}_r^{\dagger}\tilde{\mathbf{b}} + (\mathbf{I} - \mathbf{A}_r{}^{\dagger}\mathbf{A}_r)\boldsymbol{\psi}. \tag{5.16}$$

In this evaluation, the imitator robot is taken to be a 7-DoF arm with link lengths 10, 5, 5, 5, 5, 5 and $10\,cm$, and $\boldsymbol{\psi} == \boldsymbol{L}_r(\mathbf{x}_r^* - \mathbf{x})$, where $\boldsymbol{L}_r = 1$ and $\mathbf{x}_r^* = (-10°, -10°, -10°, -10°, -10°, -10°, -10°)^{\top}$. The start posture is chosen such that the initial end-effector position matches that of the demonstration (*i.e.,* $\mathbf{q} = (0°, 90°, -90°, 85°, 90°, -1°, -81.5°)^{\top}$). The resulting movement is shown in Figure 5.7-B above. As can be seen, despite the significant difference in embodiment, the task-oriented part of the movement is effectively reproduced, while the imitator-specific null space controller appropriately handles the added redundancy.

5.4.3 Real World Human Arm

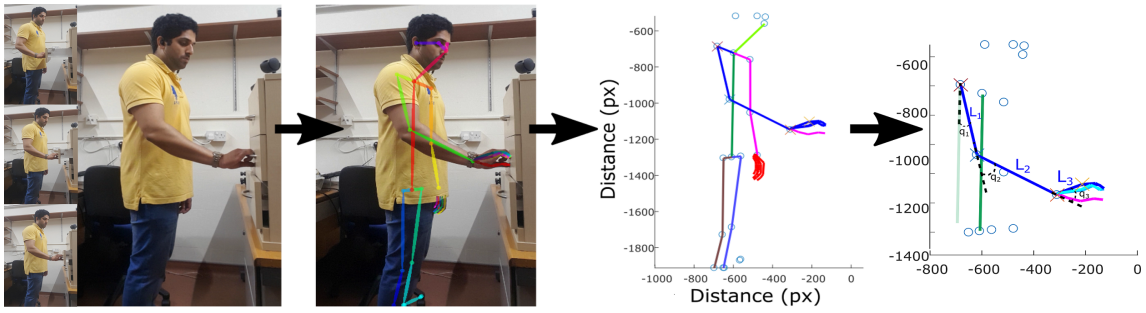


FIGURE 5.8: Sample flow of obtaining natural demonstration data from user. Video of the subject is recorded during the action of repeatedly opening and closing drawers of various heights to different lengths. After collection, the video feed is overlaid with a skeleton using Openpose [125] to obtain the positions of body parts, such that joint angles and trajectories can be extracted into Matlab. The shoulder, elbow and wrist joint angles are used to learn the constraints contained within the set of demonstrations from a set of candidate constraints.

The aim of this final experiment is to test the performance of the proposed approach in the real world using data from a human demonstrator so that task-oriented behaviour can be retargeted onto a robotic system of a different embodiment. The setup is as follows.

The task chosen for this experiment is to teach a robotic system the skill of opening and closing a three-tiered set of drawers (see Figure 5.8). To collect data, the human demonstrator stands in front of the drawers at approximately an arm's length distance. The starting state of each drawer is randomised (varying from anywhere between fully closed to completely open). Starting with the top drawer, it is moved to a random distance in the opening or closing direction. This is repeated for each of the drawers, producing a total three trajectories which are used for learning a model. A side view of this is recorded from a single 12MP phone camera (with a sensor of 1.4μm pixels and aperture of f/1.7) placed roughly 1 m away from the demonstrator. The video data is then post-processed to overlay a skeleton using Openpose [125] to estimate the joint lengths as well as to extract the joint angles and velocities during movement. Constrained motion data of movement in the sagittal plane is gathered from three joints of the demonstrator's arm. This is done by observing flexion and

extension of the shoulder, elbow and wrist (as well as abduction and adduction of the wrist depending on the forearms pronation/supination. This yields an average of 11 frames per trajectory which translates into 11 data points.

Now that the data is collected, to set it up for learning, the joint angles of the human demonstrator are treated as the state $\mathbf{x} := \mathbf{q} \in \mathbb{R}^3$ and the joint velocities as the action $\mathbf{u} := \dot{\mathbf{q}} \in \mathbb{R}^3$. The task space is described by the end-effector coordinates $\mathbf{r} = (r_x, r_y, r_\theta)^\top$ referring to the hand position and orientation, respectively. The task constraints are described in the form of Equation 5.13, where $\mathbf{J} \in \mathbb{R}^{3\times3}$ is the manipulator Jacobian of the demonstrator. To construct the Jacobian which simulates the human demonstrator as a system, the link lengths are calculated from the skeleton in Openpose for each frame. As these can vary from frame to frame depending on obscurities in the demonstrators pose and imprecision of Openpose, the mean of the joint lengths at every frame for the 3 trajectories are used. Moreover, when translating recorded movements from pixels to the x and y axis in Matlab, joint lengths are extremely large where the upper arm measures at around 30 meters. Therefore the scale of these joint lengths are proportionally reduced to around 10cm by dividing each by 300. $\mathbf{\Lambda} \in \mathbb{R}^{3\times3}$ is the selection matrix specifying the coordinates to be constrained. In this experiment, $\mathbf{\Lambda}_{x,y}$ represents the ground truth, since the end-effector (demonstrator's hand) must be maintained at the height y of the drawer handle (y changes in each demonstration depending on the which drawer is being manipulated), and x is the task space in which opening or closing action occurs. It is assumed that the control policy in $\mathbf{w}$ is resolved by the subject with comfort in mind and that it moves towards a target joint pose following Equation 5.15 with $\mathbf{x}^* = (-90°, 90°, 0°)^\top$. This posture is chosen as it lies in the middle of each joint's optimal range following RULA, resulting in a high score according to RULA's standard ergonomic assessment procedure [120] (see Figure 5.4 further above).

Since the human demonstrations are modelled as a system with its respective Jacobian matrix, $\mathbf{u}$, $\mathbf{x}$ and $\boldsymbol{\pi}$, it can be evaluated like any other robotic system presented

so far [126, 127]. The experiment is repeated 45 times yielding a total of 135 trajectories (45 repetitions for each of the 3 drawers). This is done to verify that the performance of learning with the candidate constraints is consistent.

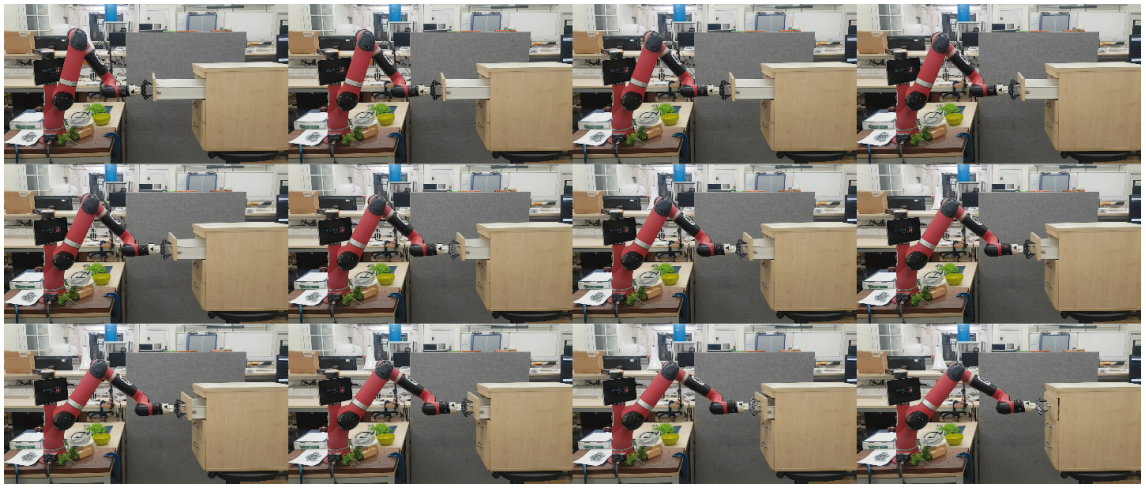


FIGURE 5.9: 3DoF Human to 7DoF Sawyer Robot task transfer

The true decomposition of the behaviour, (*i.e.,* $\mathbf{v}$ and $\mathbf{w}$) are not known, however they can be estimated using $\mathbf{\Lambda}_{x,y}$. The initial aspect that can be evaluated is the $E_{\tilde{\mathbf{w}}}$, however the variance in $\mathbf{u}$ is quite small and thus $E[\tilde{\mathbf{N}}]$ from Equation 5.4 is reported which is 6.3329 ± 5.7832(mean$\pm$s.d.). Looking at the learnt $\mathbf{\Lambda}$, the correct constraints are consistently selected using the novel approach. To demonstrate this, the learnt constraint matrix $\tilde{\mathbf{A}}$ is used to produce $\tilde{\mathbf{b}}$ and the task-oriented trajectory is reproduced on the Sawyer; a 7DoF revolute physical robotic system with a maximum reach of 1260mm and precision of $\pm$0.1mm. A *closing of a drawer* trajectory is selected from the human demonstration data. $\mathbf{J} \in \mathbb{R}^{3\times7}$ is the manipulator Jacobian where a correspondence is drawn between these and the human arm's decomposed Jacobian (as done for retargeting in Figure 5.7). The start pose is $\mathbf{q} = (-3.89°, 42.82°, 25.48°, -76.96°, -8.23°, 32.82°, 88.93°)^\top$. $\boldsymbol{\psi} == \boldsymbol{L}_r(\mathbf{x}_r^* - \mathbf{x})$, where $\boldsymbol{L}_r = 1$ and $\mathbf{x}_r^* = (-70.29°, -32.47°, -15.92°, 53.24°, -32.55°, -17.48°, -68.23°)^\top$. The resulting trajectory is presented in Figure 5.9 above. As shown, the Sawyer is able to reproduce the task space component of closing the drawer using its own

embodiment and a different $\boldsymbol{\pi}$ to resolve redundancy subject to the same task constraints.

5.5 Conclusion

In this chapter, a method based on programming by demonstration is proposed to learn null space policies from constrained motion data. It highlights how humans have predictable ways of performing tasks which are captured in ergonomics. Moreover, it is shown how this predictability can be used to make assumptions on demonstration data in order to support learning of task constraints. This assumed knowledge helps extract constraints contained within the task. Once the constraints are learnt from candidate constraints, this and the relating task-oriented behaviour can be transferred, from for example a human demonstrator, to robotic systems. The main advantage to using this is the retargeting of not only the systems redundancy resolution but also the entire system itself with another of a different embodiment, which can repeat a task accurately while being subject to the same constraints. On a lesser note, this proposed approach can be used to learn directly from observed actions without the need to decompose motion data into a task and null space component.

The effectiveness of the method has first been demonstrated in a simulated toy experiment (which requires no information regarding the constraint). Then, using candidate constraints represented as a Jacobian, to learn with a more complex 3-link planar system. On top of evaluating performance, comparisons to the state-of-the-art [13, 29] have been conducted. In addition to this, demonstrations of retargeting have been presented, first, for obstacle avoidance using a system's null space component to resolve redundancy differently, and second, for retargeting from one system to another of a different embodiment. After this, the approach has been used in a real world experiment using data collected from a human demonstrator,

which is validated through task-oriented reproduction on a 7DoF physical robot. All experiments are in agreement that the constraints can be learnt from demonstration (in some cases using prior knowledge). In addition, the evaluations show that the method can (i) learn with very little data ($E_{\tilde{\mathbf{w}}}$ below 10^{-14} with just five data points) and (ii) handle noise ($E_{\tilde{\mathbf{w}}}$ below 10^{-1} with normalised additive white gaussian noise below 0.15). In the comparative experiment, the approach is shown to outperform the current state-of-the-art approach in a simulated 3DoF robot manipulator control problem where motions are reproduced using the learnt constraints. It is also used to demonstrate retargeting of a system's null space component to resolve redundancy such that an obstacle can be avoided. Moreover, retargeting from the simulated 3DoF demonstrator to a 7DoF robot imitator of different embodiment is shown. Finally, the approach is verified in a real world experiment where demonstrations from a human subject are used to consistently learn the constraint matrix (using candidate constraints), which allows for accurate decomposition of the demonstrated task space, and in turn, task-oriented reproduction on the Sawyer, a 7DoF physical robot with a different embodiment.

Chapter 6

Conclusion

In this thesis, the optimisation of redundancy in systems by learning task-oriented behaviour through programming by demonstration has been explored; for the transfer of demonstrated skills to robotic systems.

The focus of this thesis is on learning constraint models from movement data such that generalisation of the task can be achieved across systems of different embodiment. In addition to this, it also focuses on using the learnt model to allow for redundancy resolution through optimisation of its null space. This is done by replacing this null space movement with a control policy which accomplishes a secondary goal such as singularity avoidance (without interfering with the task-oriented behaviour).

The primary motivation behind this work is for the simplified use of robotic systems by naive users through programming by demonstration (see Section 2.2). This research aims to aid in the control of systems subject to uncertain constraints due to the complexity and/or naivety of non-expert users. It does it by ensuring that demonstrations may be performed in a way which is easier for the demonstrator, rather than taking the system's structure into account. This research goal is achieved through three stages where (i) the existing constraint learning algorithm is modified by optimising its search space parameters through the use of gradient descent.

This not only allows for higher DoF (DoF) systems to be evaluated with lower cost hardware but also achieves learning in significantly faster times. (ii) Then, the so–called *manipulability* analysis, first introduced by Yoshikawa [3], is embedded into the constraint learning approach. This produces a new way to learn a constrained system's manipulability using motion data. Its use is demonstrated in a system's decomposed null space for singularity avoidance through manipulability maximisation. It works with a set of given candidate constraints, which allows naive users to work with unfamiliar systems and tasks. (iii) Finally, a novel approach is formed, where ergonomic priors from human demonstrations are considered. This allows for the transfer of generalised task-oriented behaviour to systems of a different embodiment, such that it can be adapted to suit the provided system.

Chapter 3 looks at the performance of the proposed gradient descent-based learning method. It is shown how this method can be used to estimate the null space projection matrix of a kinematically constrained system in the absence of any prior knowledge regarding the underlying control policy. Applying gradient descent to the, at the time, state-of-the-art constraint learning approach reduces the search space when learning constraints. The findings reveal that this novel method is significantly faster than the state-of-the-art brute force method when handling 4D data or higher using a linear policy. Moreover, unlike the brute force approach which is limited to 4D data (due to hardware limitations), it is shown that the new method can handle both linear and non-linear data of higher dimensions of up to and including 9D data. This proposed approach makes constraint learning applicable to modern robotic systems with greater DoF (DoF) and significantly reduces the time taken to learn a constraint model. As it also has a lower demand on computation costs, this in turn expands its applicability to lower cost computers.

Chapter 4 tackles the problems relating to the control of systems subject to uncertain constraints due to the complexity and/or naivety of non-expert users. With the help of the new gradient descent-based method and its applicability to higher

dimensional data, a new approach is devised which makes use of Yoshikawa's manipulability analysis (see Section 2.4 for background on manipulability analysis). This work considers the control of systems subject to uncertain constraints from a set of candidate constraints. The new approach shows that a constrained system's manipulability can be learnt through demonstration. Then, by using the learnt manipulability as a cost function, it is possible to control a system such that it avoid singularities while performing a task. The optimised movements from the proposed approach result in an autonomous system that moves towards the goal, while handling redundancy by moving away from singular regions through local optimisation. When compared to other control policies such as a zero policy and a point attractor, the proposed approach allows for the completion of the task. On the other hand, the other policies are shown to succumb to singularities within the same task resulting in either no movement at all or unpredictable behaviour. This method is tested with avoiding singularities in the joint space as well as end-effector space, and is thus applicable to handle manipulability maximisation in either. Moreover, tests are performed in the real world teaching a 7DoF robot in a drawer closing task with a linear constraint (see Section 4.6.3) as well as a reaching task with a non-linear constraint (see Section 4.8.3), in the joint space and end-effector space, respectively.

Chapter 5 presents a new method for programming by demonstration whereby explicit use of the underlying null space control policy—as determined by humans' stereotypical or ergonomic features—is used to learn the task and null spaces involved in the behaviour and their underlying constraints. The approach accounts for differences in human ergonomics and robot manipulability where both the demonstrator and learner may have different optimal poses, *i.e.,* what is ergonomic for a human may lead to a singular posture in a robot. This approach uses knowledge from ergonomic literature to form an estimate of how humans resolve redundancy which can be used to decompose task-oriented motions for its transfer to robotic systems. The focus of this work lies in making it as natural as possible for a human to demonstrate a task, as it cannot be expected for naive demonstrators to have any

understanding of handling robots. It plays a significant role for use by naive users as demonstrations are performed by a human without having to directly interact with the robotic system. The main advantage to using this is the retargeting of not only the system's redundancy resolution but also the entire system itself with another of a different embodiment (which can repeat a task accurately while being subject to the same constraints). Following this, a pipeline is presented which takes natural demonstrations from a human and generalises the task-oriented behaviour for its execution on a robotic system of a different embodiment. This proposed approach can be used to learn directly from observed actions without the need to decompose motion data into a task and null space component unlike the current state-of-the-art approach. It is shown that the method can learn with very little data and it is also used to demonstrate retargeting of a system's null space component to resolve redundancy such that an obstacle can be avoided. Moreover, retargeting through the learnt constraints (from a set of candidate constraints) from the simulated 3DoF demonstrator to a 7DoF robot imitator of different embodiment is shown. Finally, the approach is verified in a real world experiment in a drawer opening/closing task. In this, demonstrations from a human subject are used to consistently learn the constraints through estimation of a selection matrix, which allows for accurate decomposition of the demonstrated task space and task-oriented reproduction on the Sawyer, a 7DoF physical robot with a different embodiment.

6.1 Future Work

As this work primarily focuses on aiding naive users, future work considers the need for empirical work with a group of said users. With respect to singularity avoidance, it will be interesting to see how well naive users can teach systems through programming by demonstration. It would also be of interest to consider tasks which are prone to encountering singularities, under the assumption that users do not know how to deal with such singularities. Following this, perhaps a general guideline could be proposed for users to follow, in order to achieve sufficient variability in demonstrated tasks for robust learning.

With the latest findings where task and null spaces are learnt based on stereotypical features of demonstrators' posture control, it would be useful to have a large study with naive subjects to see how robust the method is. It would be interesting to consider this under demonstrations coming from humans of different embodiment, *i.e.,* heights and joint lengths. On a similar stream, it would also be interesting to have a more in-depth look at the so called *stereotypical features*, especially in cases where subjects may exhibit different consistent traits which diverge greatly from typical ergonomic postures, such as due to a disability or the nature of the task being performed. Thereby, an assessment can be made on how adaptable this approach is to optimisations that follow deviated a version of RULA as well as completely different measures. A study could also focus on evaluating how well learning is achieved when healthy users exhibit minor habits in their motions (*i.e.,* where users are not instructed in any way on how to perform the task). This may end up with motions that diverge from the norm according to ergonomic standards, where such results could then be compared a group where users are taught to follow some standardised form of risk assessment. Similarly, while RULA was used to determine optimal postures, alternative risk assessment methods can be considered, which may not stand out due to various reasons such as lacking quantities of existing high quality studies, but are also applicable. This could help determine whether these

approaches can be useful in decomposing task and null spaces to further simplify the demonstration or learning procedure for naive users. Moreover, this research involving constraints could be extended to look at different applications such as exploiting environmental constraints, such as sliding the arm along a table which would the *optimal pose* of the used posture measure.

Appendix A

Evaluation Criteria for Constraint Learning

Learning the constrained manipulability may not always be a trivial matter depending on the task at hand. Factors such as high dimensionality of a system or the structure of particular constraints (when being compared to others) can lead to poor learning performance. Thus, it is necessary to define a metric to assess learning performance. Several methods exist for evaluating whether the learnt constraint is indeed the correct one affecting the system in a task.

A.1 Normalised Projected Policy Error (NPPE)

NPPE measures the difference between the policy under the real constraints and the policy under the estimated constraints [13]. NPPE in this case is defined as:

$$E_{PPE} = \frac{1}{\mathcal{N}\sigma_{\mathbf{u}}^2} \sum_{n=1}^{\mathcal{N}} ||\mathbf{N}\boldsymbol{\pi}_n - \tilde{\mathbf{N}}\boldsymbol{\pi}_n||^2 \tag{A.1}$$

where $\mathcal{N}$ is the number of data points, $\boldsymbol{\pi}_n$ are samples from the policy, $\mathbf{N}$ is the true projection matrix and $\tilde{\mathbf{N}}$ is the learnt projection matrix. The error is normalised by the variance of the observations under the true constraints $\sigma^2_{\mathbf{u}}$. This method is only used for validating the results, as it requires the ground truth $\boldsymbol{\pi}_n$ and $\mathbf{N}$.

A.2 Normalised Projected Observation Error (NPOE)

To evaluate the projection, the more accurate NPPE method could be used [23], however it requires $\boldsymbol{\pi}_n$ and $\mathbf{N}$ which may not be available as learning the nullspace in a practical aspect is performed without prior knowledge of the $\boldsymbol{\pi}_n$ or $\mathbf{N}$, thus the NPOE is used instead:

$$E_{POE} = \frac{1}{\mathcal{N}\sigma^2_{\mathbf{u}}} \sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\mathbf{N}}\mathbf{u}_n||^2 \tag{A.2}$$

Bibliography

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[2] M. Howard, D. Braun, and S. Vijayakumar, "Transferring human impedance behavior to heterogeneous variable impedance actuators," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 847–862, 2013.

[3] T. Yoshikawa, "Analysis and control of robot manipulators with redundancy," in *Robotics research: the first international symposium.* Mit Press Cambridge, MA, USA, 1984, pp. 735–747.

[4] A. Hussein, M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys*, vol. 50, 04 2017.

[5] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation." *Philos Trans R Soc Lond B Biol Sci*, vol. 358, no. 1431, pp. 537–547, March 2003.

[6] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Robot Programming by Demonstration.* Springer, 01 2008, pp. 1371–1394.

[7] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233 – 242, 1999.

[8] ——, "Learning from demonstration," in *Proceedings of the 9th International Conference on Neural Information Processing Systems*, ser. NIPS'96. Cambridge, MA, USA: MIT Press, 1996, p. 1040–1046.

[9] P. Bakker and Y. Kuniyoshi, "Robot see, robot do : An overview of robot imitation," *AISB96 Workshop on Learning in Robots and Animals*, 05 1996.

[10] S. Raza, S. Haider, and M.-A. Williams, "Teaching coordinated strategies to soccer robots via imitation," *IEEE International Conference on Robotics and Biomimetics*, pp. 1434–1439, 12 2012.

[11] M. Howard, S. Klanke, M. Gienger, C. Goerick, and S. Vijayakumar, "A novel method for learning policies from variable constraint data," *Autonomous Robots*, vol. 27, no. 2, pp. 105–121, 2009.

[12] M. Howard, "Learning control policies from constrained motion," Ph.D. dissertation, University of Edinburgh, 2009.

[13] H. C. Lin, P. Ray, and M. Howard, "Learning task constraints in operational space formulation," in *IEEE Int. Conf. Robotics & Automation*, 2017, pp. 309–315.

[14] C. Eppner and O. Brock, "Planning grasp strategies that exploit environmental constraints," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 4947–4952.

[15] J. Bimbo, E. Turco, M. Ghazaei Ardakani, M. Pozzi, G. Salvietti, V. Bo, M. Malvezzi, and D. Prattichizzo, "Exploiting robot hand compliance and environmental constraints for edge grasps," *Frontiers in Robotics and AI*, vol. 6, p. 135, 2019.

[16] V. Babin and C. Gosselin, "Picking, grasping, or scooping small objects lying on flat surfaces: A design approach," *The International Journal of Robotics Research*, vol. 37, no. 12, pp. 1484–1499, 2018.

[17] C. Eppner, R. Deimel, J. Álvarez Ruiz, M. Maertens, and O. Brock, "Exploitation of environmental constraints in human and robotic grasping," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 1021–1038, 2015.

[18] M. Howard, S. Klanke, M. Gienger, C. Goerick, and S. Vijayakumar, "Robust constraint-consistent learning," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2009, pp. 4629–4636.

[19] F. Udwadia and R. Kalaba, *Analytical Dynamics: A New Approach*. Cambridge University Press, 2007.

[20] N. Vahrenkamp, H. Arnst, M. Wächter, D. Schiebener, P. Sotiropoulos, M. Kowalik, and T. Asfour, "Workspace analysis for planning human-robot interaction tasks," in *IEEE Int. Conf. Humanoid Robots*, 2016, pp. 1298–1303.

[21] J. Manavalan and M. Howard, "Learning singularity avoidance," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.

[22] K. Tahara, S. Arimoto, M. Sekimoto, and Z.-W. Luo, "On control of reaching movements for musculo-skeletal redundant arm model," *Applied Bionics and Biomechanics*, vol. 6, no. 1, pp. 11–26, 2009.

[23] H.-C. Lin, M. Howard, and S. Vijayakumar, "Learning null space projections," in *IEEE Int. Conf. Robotics & Automation*, 2015, pp. 2613–2619.

[24] R. M. Murray, S. S. Sastry, and L. Zexiang, *A Mathematical Introduction to Robotic Manipulation*, 1st ed. USA: CRC Press, Inc., 1994.

[25] H. Cruse and M. Brüwer, "The human arm as a redundant manipulator: The control of path and joint angles," *Biological cybernetics*, vol. 57, pp. 137–44, 02 1987.

[26] S. Klanke, S. Vijayakumar, and S. Schaal, "A library for locally weighted projection regression," *Journal of Machine Learning Research*, vol. 9, no. Apr, pp. 623–626, 2008.

[27] S. Hak, N. Mansard, O. Stasse, and J. P. Laumond, "Reverse control for humanoid robot task recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 6, pp. 1524–1537, Dec 2012.

[28] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *IEEE-RAS International Conference on Humanoid Robots*, Nov 2012, pp. 323–329.

[29] C. Towell, M. Howard, and S. Vijayakumar, "Learning nullspace policies," in *IEEE Int. Conf. Intel. Robots & Sys.*, 2010, pp. 241–248.

[30] L. Armesto, J. Bosga, V. Ivan, and S. Vijayakumar, "Efficient learning of constraints and generic null space policies," in *IEEE Int. Conf. Robotics & Automation*, 2017, pp. 1520–1526.

[31] J. Manavalan and M. Howard, "Learning null space projections fast," in *Euro. Symp. Art. Neural Networks*, 2017.

[32] J. Silverio, S. Calinon, L. Rozo, and D. G. Caldwell, "Learning task priorities from demonstrations," *Trans. Rob.*, vol. 35, no. 1, p. 78–94, Feb. 2019.

[33] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent Service Robotics*, vol. 9, 09 2015.

[34] ——, *Robot Learning with Task-Parameterized Generative Models*. Springer Proceedings in Advanced Robotics, 01 2018, pp. 111–126.

[35] R. Bridger, *Introduction to Ergonomics*. CRC Press, Inc., 08 2008.

[36] G. Salvendy, *Handbook of Human Factors and Ergonomics*. USA: John Wiley & Sons, Inc., 2005.

[37] M. Robertson, Y.-H. Huang, M. O'Neill, and L. Schleifer, "Flexible workspace design and ergonomics training: Impacts on the psychosocial work environment, musculoskeletal health, and work effectiveness among knowledge workers," *Applied ergonomics*, vol. 39, pp. 482–94, 08 2008.

[38] I. Nunes and P. Bush, *Work-Related Musculoskeletal Disorders Assessment and Prevention*. IntechOpen, 04 2012, pp. 1–30.

[39] B. Bernard, S. Sauter, L. Fine, M. Petersen, and T. Hales, "Job task and psychosocial risk factors for work-related musculoskeletal disorders among newspaper employees," *Scandinavian Journal of Work, Environment & Health*, vol. 20, no. 6, pp. 417–426, 1994.

[40] P. Buckle, "Upper limb disorders and work: The importance of physical and psychosocial factors," *Journal of Psychosomatic Research*, vol. 43, no. 1, pp. 17 – 25, 1997, workplace.

[41] H. &. S. E. (HSE-UK), "Work-related musculoskeletal disorder (wrmsds) statistics, great britain, 2015," Great Britain, 2015.

[42] ——, "Work-related musculoskeletal disorder (wrmsds) statistics in great britain, 2019," Great Britain, 2019.

[43] B.-T. Karsh, F. Moro, and M. Smith, "The efficacy of workplace ergonomic interventions to control musculoskeletal disorders: A critical analysis of the peer-reviewed literature," *Theoretical Issues in Ergonomics Science*, vol. 2, pp. 23–96, 01 2001.

[44] S. Brewer, D. Van Eerd, B. Amick, E. Irvin, K. Daum, F. Gerr, J. Moore, K. Cullen, and D. Rempel, "Workplace interventions to prevent musculoskeletal and visual symptoms and disorders among computer users: A systematic review," *Journal of occupational rehabilitation*, vol. 16, pp. 325–58, 10 2006.

[45] M. Calnan, "Musculoskeletal Disorders and the Workplace: Low Back and Upper Extremities. Panel on Musculoskeletal Disorders and the Workplace, Commission on Behavioral and Social Sciences and Education, National Research Council and Institute of Medicine. Washington DC: National Academy Press, 2001, pp. 429, £39.95. ISBN: 309-07284-0 (HB)." *International Journal of Epidemiology*, vol. 31, no. 3, pp. 702–702, 06 2002.

[46] L. Punnett and D. Wegman, "Work-related musculoskeletal disorders: The epidemiologic evidence and the debate," *Journal of electromyography and kinesiology : official journal of the International Society of Electrophysiological Kinesiology*, vol. 14, pp. 13–23, 03 2004.

[47] R. Westgaard and J. Winkel, "Ergonomic intervention research for improved musculoskeletal health: A critical review," *International Journal of Industrial Ergonomics*, vol. 20, no. 6, pp. 463 – 500, 1997.

[48] G. Borg, *Physical performance and perceived exertion*, ser. Studia psychologica et paedagogica Ser altera Investigationes. C. W. K. Gleerup, 1962.

[49] ——, "Perceived exertion as an indicator of somatic stress." *Scandinavian journal of rehabilitation medicine*, vol. 2 2, pp. 92–8, 1970.

[50] R. Robertson, F. Goss, J. Rutkowski, B. Lenz, C. Dixon, J. Timmer, K. Frazee, J. Dubé, and J. Andreacci, "Concurrent validation of the omni perceived exertion scale for resistance exercise," *Medicine and science in sports and exercise*, vol. 35, pp. 333–41, 02 2003.

[51] G. David, "Ergonomic methods for assessing exposure to risk factors for work-related musculoskeletal disorders," *Occupational medicine (Oxford, England)*, vol. 55, pp. 190–9, 06 2005.

[52] E. Vieira and S. Kumar, "Working postures: A literature review," *Journal of occupational rehabilitation*, vol. 14, pp. 143–59, 07 2004.

[53] D. Chaffin, "Localized muscle fatigue - definition and measurement," *Journal of Occupational and Environmental Medicine*, vol. 15, pp. 346–354, 04 1973.

[54] O. Karhu, P. Kansi, and I. Kuorinka, "Correcting working postures in industry: A practical method for analysis," *Applied Ergonomics*, vol. 8, no. 4, pp. 199 – 201, 1977.

[55] E.-P. Takala, P. Irmeli, M. Forsman, G.-rA. Hansson, M. Svend Erik, W. Neumann, G. Sjøgaard, K. Veiersted, R. Westgaard, and J. Winkel, *Systematic evaluation of observational methods assessing biomechanical exposures at work*. International Ergonomics Association, 2009.

[56] M. Gómez-Galán, J. Pérez-Alonso, A. J. Callejon-Ferre, and J. López-Martínez, "Musculoskeletal disorders: Owas review," *Industrial health*, vol. 55, 05 2017.

[57] L. Mcatamney and E. N. Corlett, "Rula: A survey method for the investigation of work-related upper limb disorders," *Applied ergonomics*, vol. 24, pp. 91–9, 05 1993.

[58] S. Dockrell, E. O'Grady, K. Bennett, C. Mullarkey, R. M. Connell, R. Ruddy, S. Twomey, and C. Flannery, "An investigation of the reliability of rapid upper limb assessment (rula) as a method of assessment of children's computing posture," *Applied Ergonomics*, vol. 43, no. 3, pp. 632 – 636, 2012.

[59] S. Namwongsa, R. Puntumetakul, M. S. Neubert, S. Chaiklieng, and R. Boucaut, "Ergonomic risk assessment of smartphone users using the rapid upper limb assessment (rula) tool," in *PloS one*, 2018.

[60] K. H. j. Kim Jae young, Choi Jae wook, "The relation between work-related musculoskeletal symptoms and rapid upper limb assessment(rula) among vehicle assembly workers." *Korean J Prev Med*, vol. 32, no. 1, pp. 48–59, 1999.

[61] M. Massaccesi, A. Pagnotta, A. Soccetti, M. Masali, C. Masiero, and F. Greco, "Investigation of work-related disorders in truck drivers using rula method," *Applied ergonomics*, vol. 34, pp. 303–7, 08 2003.

[62] Y.-K. Kong, S. yong Lee, K.-S. Lee, and D.-M. Kim, "Comparisons of ergonomic evaluation tools (alla, rula, reba and owas) for farm work," *International Journal of Occupational Safety and Ergonomics*, vol. 24, no. 2, pp. 218–223, 2018, pMID: 28301984.

[63] S. Hignett and L. Mcatamney, "Rapid entire body assessment (reba)," *Applied ergonomics*, vol. 31, pp. 201–5, 05 2000.

[64] M. Sonne, D. L. Villalta, and D. M. Andrews, "Development and evaluation of an office ergonomic risk checklist: Rosa – rapid office strain assessment," *Applied Ergonomics*, vol. 43, no. 1, pp. 98 – 108, 2012.

[65] A. Coyle, "Comparison of the rapid entire body assessment and the new zealand manual handling 'hazard control record', for assessment of manual handling hazards in the supermarket industry." *Work*, vol. 24 2, pp. 111–6, 2005.

[66] D. A. Madani and A. Dababneh, "Rapid entire body assessment: A literature review," *American Journal of Engineering and Applied Sciences*, vol. 9 1, pp. 107–118, 2016.

[67] M. Sonne and D. M. Andrews, "The rapid office strain assessment (rosa): Validity of online worker self-assessments and the relationship to worker discomfort," *Occupational Ergonomics*, vol. 10, no. 3, pp. 83–101, 2011.

[68] U.S. Department of Health and Human Services, "Work practices guide for manual lifting," National Institute for Occupational Safety and Health, Tech. Rep. 81-122, March 1981.

[69] T. R. Waters, V. Putz-Anderson, A. Garg, and L. J. Fine, "Revised niosh equation for the design and evaluation of manual lifting tasks," *Ergonomics*, vol. 36, no. 7, pp. 749–776, 1993.

[70] P. G. Dempsey, "Usability of the revised niosh lifting equation," *Ergonomics*, vol. 45, no. 12, pp. 817–828, 2002.

[71] T. R. Waters, S. L. Baron, L. A. Piacitelli, V. P. Anderson, T. Skov, M. Haring-Sweeney, D. K. Wall, and L. J. Fine, "Evaluation of the revised niosh lifting equation. a cross-sectional epidemiologic study." *Spine*, vol. 24 4, pp. 386–95, 1999.

[72] P. G. Dempsey, "A survey of lifting and lowering tasks," *International Journal of Industrial Ergonomics*, vol. 31, no. 1, pp. 11 – 16, 2003.

[73] T. Waters, S. Baron, and K. Kemmlert, "Accuracy of measurements for the revised niosh lifting equation," *Applied Ergonomics*, vol. 29, no. 6, pp. 433 – 438, 1998.

[74] M.-L. Lu, T. R. Waters, E. Krieg, and D. Werren, "Efficacy of the revised niosh lifting equation to predict risk of low-back pain associated with manual lifting: A one-year prospective study," *Human Factors*, vol. 56, no. 1, pp. 73–85, 2014.

[75] J. S. Moore and A. Garg, "The strain index: A proposed method to analyze jobs for risk of distal upper extremity disorders," *American Industrial Hygiene Association Journal*, vol. 56, no. 5, pp. 443–458, 1995.

[76] E. Occhipinti, "Ocra: a concise index for the assessment of exposure to repetitive movements of the upper limbs." *Ergonomics*, vol. 41 9, pp. 1290–311, 1998.

[77] G. Li and P. Buckle, "A practical method for the assessment of work-related musculoskeletal risks - quick exposure check (qec)," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 42, 10 1998.

[78] G. David, P. Buckle, V. Woods, 'Great Britain. H. & S. E (HSE-UK)', and 'University of Surrey. Robens Centre for Health Ergonomics', "Further development of the usability and validity of the quick exposure check (qec)," Great Britain, 2005.

[79] D. Kee and W. Karwowski, "Luba: An assessment technique for postural loading on the upper body based on joint motion discomfort and maximum holding time," *Applied Ergonomics*, vol. 32, pp. 357–366, 09 2001.

[80] A. Sanchez-Lite, M. Garcia, R. Domingo, and M. Sebastián, "Novel ergonomic postural assessment method (nerpa) using product-process computer aided engineering for ergonomic workplace design," *PloS one*, vol. 8, p. e72703, 08 2013.

[81] B. Juul-Kristensen, N. Fallentin, and C. Ekdahl, "Criteria for classification of posture in repetitive work by observation methods: A review," *International Journal of Industrial Ergonomics*, vol. 19, pp. 397–411, 05 1997.

[82] S. Yazdanirad, A. Khoshakhlagh, E. Habibi, A. Zare, M. Zeinodini, and F. Dehghani, "Comparing the effectiveness of three ergonomic risk assessment methods—rula, luba, and nerpa—to predict the upper extremity musculoskeletal disorders," *Indian Journal of Occupational and Environmental Medicine*, vol. 22, p. 17, 01 2018.

[83] T. Jones and S. Kumar, "Comparison of ergonomic risk assessment output in four sawmill jobs," *International journal of occupational safety and ergonomics : JOSE*, vol. 16, pp. 105–11, 01 2010.

[84] M.-È. Chiasson, D. Imbeau, K. Aubry, and A. Delisle, "Comparing the results of eight methods used to evaluate risk factors associated with musculoskeletal

disorders," *International Journal of Industrial Ergonomics*, vol. 42, p. 478–488, 09 2012.

[85] P. Drinkaus, R. Sesek, D. Bloswick, T. Bernard, B. Walton, B. Joseph, G. Reeve, and J. Counts, "Comparison of ergonomic risk assessment outputs from rapid upper limb assessment and the strain index for tasks in automotive assembly plants," *Work (Reading, Mass.)*, vol. 21, pp. 165–72, 02 2003.

[86] D. Roman-Liu, "Comparison of concepts in easy-to-use methods for msd risk assessment," *Applied ergonomics*, vol. 45, 07 2013.

[87] A. Shafti, A. Ataka, B. Urbistondo Lazpita, A. Shiva, H.A. Wurdemann and K. Althoefer, "Real-time robot-assisted ergonomics," in *IEEE Int. Conf. Robotics & Automation*, 2019.

[88] W. Khalil and E. Dombre, *Modeling, Identification and Control of Robots*, W. Khalil and E. Dombre, Eds. Oxford: Butterworth-Heinemann, 2002.

[89] G. Chen, L. Zhang, Q. Jia, and H. Sun, "Singularity analysis of redundant space robot with the structure of canadarm2," *Mathematical Problems in Engineering*, vol. 2014, pp. 1–9, 06 2014.

[90] D. N. Oetomo, "Singularity analysis and handling towards mobile manipulation," Ph.D. dissertation, National University of Singapore, 2004.

[91] S. Chiaverini, G. Oriolo, and I. Walker, *Kinematically Redundant Manipulators*. Springer, 01 2008, pp. 245–268.

[92] D. L. Pieper, "The kinematics of manipulators under computer control," Ph.D. dissertation, Stanford University, 1969.

[93] B. Roth, "Performance evaluation of manipulators from a kinematic viewpoint," *Cours de Robotique, IRIA*, pp. 233–263, 1976.

[94] H.-Y. Lee and C.-G. Liang, "Displacement analysis of the general spatial 7-link 7r mechanism," *Mechanism and Machine Theory*, vol. 23, no. 3, pp. 219 – 226, 1988.

[95] M. Raghavan and B. Roth, "Inverse Kinematics of the General 6R Manipulator and Related Linkages," *Journal of Mechanical Design*, vol. 115, no. 3, pp. 502–508, 09 1993.

[96] Y. Yang, V. Ivan, Z. Li, M. Fallon, and S. Vijayakumar, "iDRM: Humanoid motion planning with realtime end-pose selection in complex environments," in *IEEE Int. Conf. Humanoid Robots*, 2016, pp. 271–278.

[97] A. M. Sundaram, O. Porges, and M. A. Roa, "Planning realistic interactions for bimanual grasping and manipulation," in *IEEE-RAS Int. Conf on Humanoids*, 2016, pp. 987–994.

[98] N. Vahrenkamp, T. Asfour, G. Metta, G. Sandini, and R. Dillmann, "Manipulability analysis," in *IEEE Int. Conf. Humanoid Robots*, 2012, pp. 568–573.

[99] N. Vahrenkamp and T. Asfour, "Representing the robot's workspace through constrained manipulability analysis." *Auton. Robots*, vol. 38, no. 1, pp. 17–30, 2015.

[100] S. Chiaverini, B. Siciliano, and O. Egeland, "Kinematic analysis and singularity avoidance for a seven-joint manipulator," 06 1990, pp. 2300 – 2305.

[101] B.-H. Jun, P.-M. Lee, and S. Kim, "Manipulability analysis of underwater robotic arms on rov and application to task-oriented joint configuration," *Journal of Mechanical Science and Technology*, vol. 22, pp. 887–894, 05 2008.

[102] J. Sverdrup-Thygeson, S. Moe, K. Y. Pettersen, and J. T. Gravdahl, "Kinematic singularity avoidance for robot manipulators using set-based manipulability tasks," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, 2017, pp. 142–149.

[103] S. Moe, G. Antonelli, A. R. Teel, K. Y. Pettersen, and J. Schrimpf, "Set-based tasks within the singularity-robust multiple task-priority inverse kinematics framework: General formulation, stability analysis, and experimental results," *Frontiers in Robotics and AI*, vol. 3, p. 16, 2016.

[104] M. Stilman and J. Kuffner, "Planning among movable obstacles with artificial constraints," *Int. J. Robotics Research*, vol. 27, no. 11-12, pp. 1295–1307, 2008.

[105] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," in *Experimental Robotics*. Springer, 2014, pp. 477–491.

[106] R. Fernandez, A. Vazquez, I. Payo, and A. Adan, "A comparison of tactile sensors for in-hand object location," *Sensors*, 2016.

[107] P. Gil, C. Mateo, A. Delgado, and F. Torres, "Visual/tactile sensing to monitor grasps with robot-hand for planar elastic objects," in *Int. Symp. Robotics*, 2016.

[108] V. Ortenzi, H. Lin, M. Azad, J. Kuo, R. Stolkin, and M. Mistry, "Estimation of contact constraints using kinematics," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2016.

[109] C. Gehring, C. Bellicoso, S. Coros, M. Bloesch, P. Fankhauser, M. Hutter, and R. Siegwart, "Dynamic trotting on slopes for quadrupedal robots," in *IEEE/RSJ IEEE Int. Conf. on Intelligent Robots and Systems*, 2015.

[110] R. Bellman, *Adaptive Control Processes. A Guided Tour.* Princeton, N. J.: Princeton University Press, 1961.

[111] V. Boln-Canedo, N. Snchez-Maroo, and A. Alonso-Betanzos, *Feature Selection for High-Dimensional Data*, 1st ed. Springer Publishing Company, Incorporated, 2015.

[112] S. Dai, S. Schaffert, A. Jasour, A. Hofmann, and B. Williams, "Chance constrained motion planning for high-dimensional robots," in *IEEE Int. Conf. Robotics & Automation*, 05 2019, pp. 8805–8811.

[113] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space," *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, vol. Vol. 1, 05 2000.

[114] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley Publishing Company, 1991.

[115] C. C Gordon, T. Churchill, C. E Clauser, B. Bradtmiller, J. T McConville, I. Tebbetts, and R. Walker, "Anthropometric survey of u.s. army personnel: Summary statistics, interim report for 1988," *Natick, Mass: U.S. Army Natick Research, Development and Engineering Center*, 01 1989.

[116] A. Sena and M. Howard, "Quantifying teaching behaviour in robot learning from demonstration," *Int. J. Robotics Research*, 2020.

[117] R. Ranganathan, A. Adewuyi, and F. A. Mussa-Ivaldi, "Learning to be lazy: Exploiting redundancy in a novel task to minimize movement-related effort," *Journal of Neuroscience*, vol. 33, no. 7, pp. 2754–2760, 2013.

[118] J. Soechting, C. Buneo, U. Herrmann, and M. Flanders, "Moving effortlessly in three dimensions: does donders' law apply to arm movement?" *Journal of Neuroscience*, vol. 15, no. 9, pp. 6271–6280, 1995.

[119] M. Robertson, B. Amick, K. DeRango, T. Rooney, L. Bazzani, R. Harrist, and A. Moore, "The effects of an office ergonomics training and chair intervention on worker knowledge, behavior and musculoskeletal risk," *Applied ergonomics*, vol. 40, pp. 124–35, 04 2008.

[120] M. Middlesworth, "A step-by-step guide to the rula assessment tool," 2019. [Online]. Available: https://ergo-plus.com/rula-assessment-tool-guide/

[121] Y. Zhao, A. Sena, F. Wu, and M. Howard, "A framework for teaching impedance behaviours by combining human and robot 'best practice'," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 10 2018, pp. 3010–3015.

[122] A. Alissandrakis, C. Nehaniv, and K. Dautenhahn, "Correspondence mapping induced state and action metrics for robotic imitation," *IEEE Transactions on Systems, Man, and Cybernetics: Cybernetics*, vol. 37, no. 2, pp. 299–307, 2007.

[123] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions Systems Man and Cybernetics*, vol. 7, pp. 868–871, 1977.

[124] O. Khatib, "A unified approach for motion and force control of robot manipulators: the operational space formulation," *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 1, pp. 43–53, December 1987.

[125] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields," in *arXiv preprint arXiv:1812.08008*, 2018.

[126] B. Lee, "A mouse with two optical sensors that eliminates coordinate disturbance during skilled strokes," *Human-Computer Interaction*, vol. 30, 03 2014.

[127] A. Gams and J. Lenarcic, "Humanoid arm kinematic modeling and trajectory generation," in *IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, 2006, pp. 301 – 305.